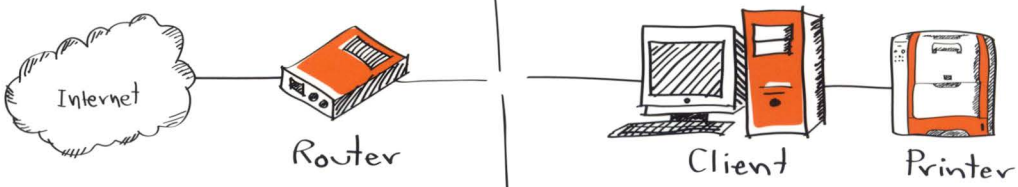
Виктор Олифер
Наталья Олифер

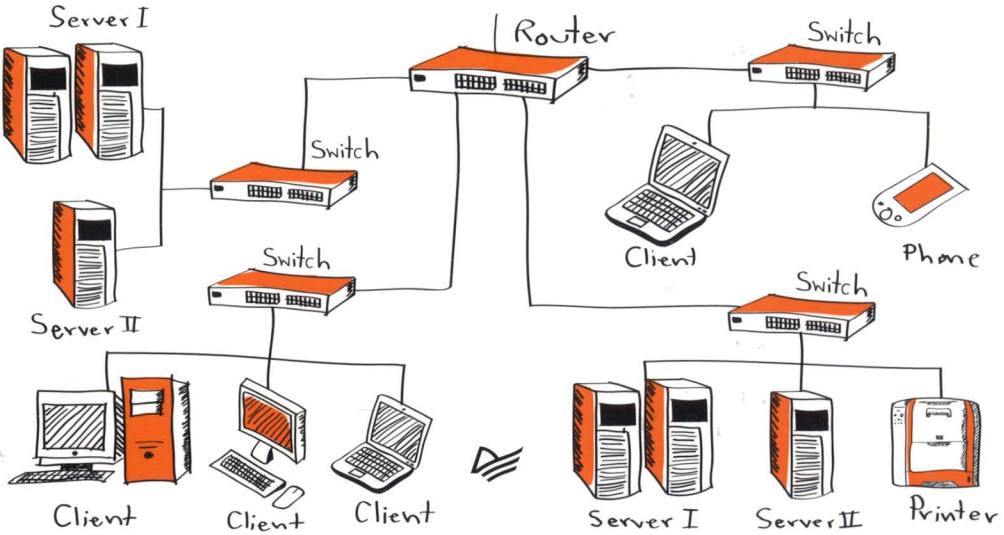


КОМПЬЮТЕРНЫЕ СЕТИ



Принципы, технологии, протоколы

Седьмое издание



Рекомендовано Министерством образования и науки РФ

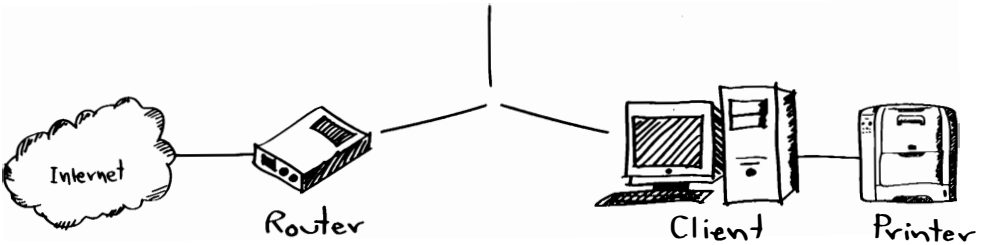
Виктор Олифер
Наталья Олифер



Компьютерные сети

Принципы, технологии, протоколы

Седьмое издание



ПИТЕР®
Санкт-Петербург • Москва • Минск
2026

ББК 32.988.02я7
УДК 004.7(07)
О-54

Олифер Виктор, Олифер Наталья

О-54 Компьютерные сети. Принципы, технологии, протоколы. 7-е изд. — СПб.: Питер, 2026. — 1104 с.: ил. — (Серия «Учебник для вузов»).

ISBN 978-5-4461-4566-9

Седьмое издание одного из лучших российских учебников по сетевым технологиям, переведенным на английский, португальский, испанский и китайский языки.

Секретом успеха этой книги, которая издается уже на протяжении более 25 лет, является постоянное и существенное изменение ее содержания, при котором материал каждого нового издания обновляется на 25-30 %, отражая новые тенденции и технологии быстро изменяющейся области компьютерных сетей. В то же время во всех изданиях сохраняется ядро, описывающее фундаментальные принципы и базовые механизмы современных сетевых технологий.

Коренной переработке подверглись материалы, посвященные беспроводной передаче данных, в частности глубоко и подробно рассматриваются технологии мобильных сотовых сетей и Wi-Fi. Также добавлены разделы о сетях дронов и спутниковых сетях Iridium и Starlink описание технологий интернета вещей, таких как ZigBee и LoRaWAN.

Материалы книги прошли успешную проверку в сложной аудитории, состоящей из слушателей с весьма различным уровнем подготовки и кругом профессиональных интересов. Среди них — студенты и аспиранты вузов, сетевые администраторы и интеграторы, начальники отделов автоматизации и преподаватели. Учебник построен так, чтобы начинающий получил основу для дальнейшего изучения, а специалист смог систематизировать и актуализировать имеющиеся знания.

Чтобы не делать книгу слишком большой, а также из-за снижения актуальности тех или иных технологий часть материалов предыдущих изданий перенесена на сайт www.olifer.co.uk.

Рекомендовано Министерством образования и науки Российской Федерации в качестве учебного пособия для студентов высших учебных заведений, обучающихся по направлению «Информатика и вычислительная техника» и по специальностям «Вычислительные машины, комплексы, системы и сети», «Автоматизированные машины, комплексы, системы и сети», «Программное обеспечение вычислительной техники и автоматизированных систем».

16+ (В соответствии с Федеральным законом от 29 декабря 2010 г. № 436-ФЗ.)

ББК 32.988.02я7
УДК 004.7(07)

Все права защищены. Никакая часть данной книги не может быть воспроизведена в какой бы то ни было форме без письменного разрешения владельцев авторских прав.

Информация, содержащаяся в данной книге, получена из источников, рассматриваемых издательством как надежные. Тем не менее, имея в виду возможные человеческие или технические ошибки, издательство не может гарантировать абсолютную точность и полноту приводимых сведений и не несет ответственности за возможные ошибки, связанные с использованием книги. В книге возможны упоминания организаций, деятельность которых запрещена на территории Российской Федерации, таких как Meta Platforms Inc., Facebook, Instagram и др. Издательство не несет ответственности за доступность материалов, ссылки на которые вы можете найти в этой книге. На момент подготовки книги к изданию все ссылки на интернет-ресурсы были действующими.

ISBN 978-5-4461-4566-9

© ООО Издательство «Питер», 2026
© Серия «Учебник для вузов», 2026
© Виктор Олифер, Наталья Олифер, 2026

Краткое содержание

ЧАСТЬ I.
ОСНОВЫ СЕТЕЙ ПЕРЕДАЧИ ДАННЫХ

Глава 1. Эволюция компьютерных сетей 27

Глава 2. Общие принципы построения сетей 45

Глава 3. Коммутация каналов и пакетов 80

Глава 4. Стандартизация и классификация сетей 107

Глава 5. Сетевые характеристики и качество обслуживания 130

Вопросы к части I 177

ЧАСТЬ II.
ТЕХНОЛОГИИ ФИЗИЧЕСКОГО УРОВНЯ ПРОВОДНЫХ СЕТЕЙ

Глава 6. Линии связи 182

Глава 7. Кодирование и мультиплексирование данных 214

Глава 8. Технология первичных сетей OTN 241

Глава 9. Технология первичных сетей DWDM 267

Вопросы к части II 281

ЧАСТЬ III.
ТЕХНОЛОГИЯ ETHERNET

Глава 10. Ethernet в локальных сетях 286

Глава 11. Отказоустойчивые и виртуальные локальные сети 325

Глава 12. Ethernet операторского класса 350

Вопросы к части III 360

ЧАСТЬ IV.
СЕТИ TCP/IP

Глава 13. Стек протоколов TCP/IP. Адресация 364

Глава 14. Протокол межсетевого взаимодействия IP 398

Глава 15. Протоколы транспортного уровня TCP и UDP 441

Глава 16. Протоколы маршрутизации и технология SDN 471

Глава 17. IPv6 как развитие стека TCP/IP 513

Вопросы к части IV 545

ЧАСТЬ V. ГЛОБАЛЬНЫЕ ПРОВОДНЫЕ КОММУНИКАЦИОННЫЕ СЕТИ

Глава 18. Организация и коммуникационные услуги глобальных сетей 555

Глава 19. Информационные услуги глобальных сетей 564

Глава 20. MPLS — транспортная технология глобальных сетей 607

Глава 21. Виртуальные частные сети 638

Вопросы к части V 654

ЧАСТЬ VI. БЕСПРОВОДНАЯ ПЕРЕДАЧА ДАННЫХ

Глава 22. Особенности беспроводных сетей 661

Глава 23. Технологии физического уровня беспроводных сетей 686

Глава 24. Сотовые и спутниковые сети 720

Глава 25. Технологии Wi-Fi и Bluetooth 783

Глава 26. Беспроводные технологии с низким энергопотреблением 820

Глава 27. Сети дронов 843

Вопросы к части VI 883

ЧАСТЬ VII. БЕЗОПАСНОСТЬ КОМПЬЮТЕРНЫХ СЕТЕЙ

Глава 28. Основные понятия и принципы информационной безопасности 891

Глава 29. Технологии аутентификации, авторизации и управления доступом 924

Глава 30. Технологии безопасности на основе анализа трафика 954

Глава 31. Атаки на транспортную инфраструктуру сети 990

Глава 32. Безопасность программного кода и сетевых служб 1022

Вопросы к части VII 1049

Ответы 1057

Рекомендуемая и использованная литература 1069

Список материалов на сайте книги www.olifer.co.uk 1072

Алфавитный указатель 1074

Оглавление

От авторов **22**
Для кого эта книга 22
Из предисловия к юбилейному шестому изданию 23
Изменения в седьмом издании 23
Благодарности 24
От издательства 24

ЧАСТЬ I.
ОСНОВЫ СЕТЕЙ ПЕРЕДАЧИ ДАННЫХ

Глава 1. Эволюция компьютерных сетей **27**
Два корня компьютерных сетей 27
Первые компьютерные сети 28
Системы пакетной обработки 28
Многотерминальные системы — прообраз сети 29
Первые глобальные сети 30
Первые локальные сети 32
Конвергенция сетей 35
Конвергенция локальных и глобальных сетей 35
Конвергенция компьютерных и телекоммуникационных сетей 36
Интернет как фактор развития сетевых технологий 38
Искусственный интеллект в компьютерных сетях 41

Глава 2. Общие принципы построения сетей **45**
Простейшая сеть из двух компьютеров 45
Совместное использование ресурсов 45
Сетевые интерфейсы 45
Связь компьютера с периферийным устройством 47
Обмен данными между двумя компьютерами 48
Доступ к периферийным устройствам через сеть 49
Сетевое программное обеспечение 50
Сетевые службы и сервисы 50
Сетевая операционная система 52
Сетевые приложения 54
Физическая передача данных по линиям связи 57
Кодирование 57
Характеристики физических каналов 59

Проблемы связи нескольких компьютеров 61
Топология физических связей 61
Адресация узлов сети 64
Коммутация 66
Обобщенная задача коммутации 67
Определение информационных потоков 67
Маршрутизация 69
Продвижение данных 72
Мультиплексирование и демультиплексирование 74
Разделяемая среда передачи данных 76

Глава 3. Коммутация каналов и пакетов 80
Коммутация каналов 80
Элементарный канал 81
Составной канал 83
Неэффективность передачи пульсирующего трафика 87
Коммутация пакетов 88
Буферизация пакетов 91
Дейтаграммная передача 92
Передача с установлением логического соединения 94
Передача с установлением виртуального канала 96
Сравнение сетей с коммутацией пакетов и каналов 98
Структура задержек в сетях с коммутацией каналов и пакетов 99
Количественное сравнение задержек. Пример 103
Ethernet — пример стандартной технологии с коммутацией пакетов 104

Глава 4. Стандартизация и классификация сетей 107
Декомпозиция задачи сетевого взаимодействия 107
Многоуровневый подход 107
Протокол и стек протоколов 109
Модель OSI 111
Цель создания модели OSI 111
Определение уровней модели OSI 112
Схема многоуровневого взаимодействия конечных узлов 115
Распределение протоколов по элементам сети 117
Вспомогательные протоколы 118
Модель OSI и сети с коммутацией каналов 120
Стандартизация сетей 121
Понятие открытой системы 121
Источники стандартов 122
Стандартизация интернета 123
Стандартные стеки коммуникационных протоколов 123
Классификация компьютерных сетей 125

Глава 5. Сетевые характеристики и качество обслуживания. 130
Типы характеристик 130
Субъективные оценки качества 130
Требования к характеристикам со стороны пользователя и поставщика услуг 131
Долговременные, среднесрочные и краткосрочные характеристики. 131
Соглашение об уровне обслуживания 132
Производительность и надежность сети 133
Идеальная и реальная сети 133
Статистические оценки характеристик сети 135
Активные и пассивные измерения в сети. 138
Характеристики задержек пакетов 141
Характеристики скорости передачи 143
Характеристики надежности сети 145
Характеристики сети поставщика услуг 146
Приложения и качество обслуживания 148
Степень равномерности порождаемого трафика. 148
Чувствительность приложений к задержкам пакетов 149
Чувствительность приложений к потерям и искажениям пакетов 150
Методы обеспечения качества обслуживания 151
Управление очередями 153
Анализ очередей 153
Очереди и различные классы трафика. 156
Техника управления очередями. 157
Механизмы кондиционирования трафика 162
Обратная связь для предотвращения перегрузок 165
Резервирование ресурсов 168
Процедура резервирования пропускной способности 169
Обеспечение заданного уровня задержек. 171
Инжиниринг трафика 172
Недостатки традиционных методов маршрутизации 172
Методы инжиниринга трафика 173
Работа в недогруженном режиме 175

Вопросы к части I 177

ЧАСТЬ II.
ТЕХНОЛОГИИ ФИЗИЧЕСКОГО УРОВНЯ ПРОВОДНЫХ СЕТЕЙ

Глава 6. Линии связи 182
Классификация линий связи 182
Первичные сети, линии и каналы связи 182
Физическая среда передачи данных 183
Аппаратура передачи данных 185

Характеристики линий связи 186
Спектральное представление сигнала 186
Затухание и опорная мощность 192
Полоса пропускания 197
Помехи 198
Пропускная способность 201
Влияние способа кодирования на пропускную способность 203
Соотношение полосы пропускания и пропускной способности 205
Проводные линии связи 206
Экранированная и неэкранированная витая пара 206
Коаксиальный кабель 208
Волоконно-оптический кабель 208
Структурированная кабельная система зданий 212

Глава 7. Кодирование и мультиплексирование данных 214
Виды кодирования 214
Кодирование дискретной информации 215
Этапы кодирования 215
Спектр информационного сигнала 216
Выбор способа кодирования 218
Кодирование дискретной информации дискретными сигналами 219
Потенциальный код NRZ 219
Биполярное кодирование AMI 220
Потенциальный код NRZI 221
Биполярный импульсный код 221
Манчестерский код 221
Избыточные коды 222
Кодирование дискретной информации аналоговыми сигналами 225
Обнаружение и коррекция ошибок 230
Методы обнаружения ошибок 230
Методы коррекции ошибок 231
Кодирование аналоговой информации 232
Кодирование аналоговой информации аналоговыми сигналами 232
Кодирование аналоговой информации дискретными сигналами 233
Мультиплексирование и коммутация 235
Мультиплексирование и коммутация на основе методов FDM и WDM 236
Мультиплексирование и коммутация на основе метода TDM 237

Глава 8. Технология первичных сетей OTN 241
Особенности первичных сетей 241
Топология и типы оборудования сетей синхронного TDM 242
Статичность нагрузки. Иерархия скоростей 245
Функции мультиплексора 246
Технологии PDH и SDH 249

Технология OTN 253
Причины создания сетей OTN 253
Иерархия скоростей 254
Функциональные уровни OTN 255
Формат кадра OTN 256
Отображение и выравнивание пользовательских данных 258
Мультиплексирование блоков OTN 260

Глава 9. Технология первичных сетей DWDM 267
Принцип работы 267
Частотные планы 270
Оборудование и топологии сетей DWDM 271
Терминальные мультиплексоры 272
Оптические мультиплексоры ввода-вывода 274
Волоконно-оптические усилители 275
Устройства компенсации дисперсии 276
Ячеистая топология и реконфигурируемые оптические кросс-коннекторы 276

Вопросы к части II 281

ЧАСТЬ III.
ТЕХНОЛОГИЯ ETHERNET

Глава 10. Ethernet в локальных сетях 286
Первый этап — разделяемая среда 286
Стандартная топология и разделяемая среда 286
Уровни Ethernet 288
MAC-адреса 290
Форматы кадров технологии Ethernet 291
Доступ к среде и передача данных 292
Возникновение и распознавание коллизии 293
Физические стандарты 10M Ethernet 296
Коммутируемый Ethernet 298
Мост как предшественник и функциональный аналог коммутатора 298
Коммутаторы 305
Скоростные версии Ethernet 311
Fast Ethernet 314
Gigabit Ethernet 315
10G Ethernet 318
100G и 40G Ethernet 319
Первое поколение 200G и 400G Ethernet 320
Новые поколения: 400G (4 линии), 800G и 1,6T Ethernet 323

Глава 11. Отказоустойчивые и виртуальные локальные сети 325
Алгоритм покрывающего дерева 325
Протокол STP 326

Фильтрация трафика 331
Агрегирование линий связи в локальных сетях 333
Транки и логические каналы 333
Динамическое агрегирование линий связи в стандарте IEEE Link Aggregation 335
Виртуальные локальные сети 338
Назначение виртуальных сетей 339
Создание виртуальных сетей на базе одного коммутатора 341
Создание виртуальных сетей на базе нескольких коммутаторов 341
Конфигурирование VLAN 343
Автоматизация конфигурирования VLAN 346
Альтернативные маршруты в виртуальных локальных сетях 347
Ограничения коммутаторов 348

Глава 12. Ethernet операторского класса 350
Движущие силы экспансии Ethernet 350
Области улучшения Ethernet 351
Разделение адресных пространств пользователей и провайдера 351
Маршрутизация, инжиниринг трафика и отказоустойчивость 352
Функции эксплуатации, администрирования и обслуживания 352
Функции OAM в Ethernet операторского класса 353
Протокол CFM 353
Протокол мониторинга качества соединений Y.1731 356
Стандарт тестирования физического соединения Ethernet 356
Интерфейс локального управления Ethernet 357
Мосты провайдера 357

Вопросы к части III 360

ЧАСТЬ IV.
СЕТИ TCP/IP

Глава 13. Стек протоколов TCP/IP. Адресация 364
Стек протоколов TCP/IP 364
Межсетевое взаимодействие 368
Типы адресов стека TCP/IP 369
Формат IP-адреса 371
Классы IP-адресов 372
Особые IP-адреса 374
Использование масок при IP-адресации 375
Порядок назначения IP-адресов 376
Централизованное распределение адресов 376
Технология бесклассовой маршрутизации CIDR 378
Отображение IP-адресов на локальные адреса 379
Протокол ARP 380
Протокол Proxy-ARP 384

Доменная служба имен DNS . . . 385
Пространство DNS-имен . . . 385
Сервер, клиент и протокол DNS . . . 388
Иерархическая организация службы DNS . . . 388
Итеративная и рекурсивная процедуры разрешения имени . . . 390
Корневые серверы . . . 391
Обратная зона . . . 392
Протокол DHCP . . . 393
Режимы DHCP . . . 393
Динамическое назначение адресов . . . 395

Глава 14. Протокол межсетевого взаимодействия IP . . . 398
IP-пакет . . . 398
Схема IP-маршрутизации . . . 401
Упрощенная таблица маршрутизации . . . 403
Таблицы маршрутизации конечных узлов . . . 404
Просмотр таблиц маршрутизации без масок . . . 406
Примеры таблиц маршрутизации разных форматов . . . 406
Источники и типы записей в таблице маршрутизации . . . 411
Пример IP-маршрутизации без масок . . . 412
Маршрутизация с использованием масок . . . 416
Структуризация сети масками одинаковой длины . . . 416
Просмотр таблиц маршрутизации с учетом масок . . . 419
Использование масок переменной длины . . . 420
CIDR и маршрутизация . . . 423
Фрагментация IP-пакетов . . . 424
Параметры фрагментации . . . 425
Механизм фрагментации . . . 426
Протокол ICMP . . . 428
Формат, типы и коды ICMP-сообщений . . . 429
Ошибка недостижимости узла и утилита traceroute . . . 431
Сообщения эхо-запрос и эхо-ответ в утилите ping . . . 433
Мобильный IPv4 . . . 434
Поддержка QoS протоколом IP . . . 437

Глава 15. Протоколы транспортного уровня TCP и UDP . . . 441
Мультиплексирование и демультиплексирование приложений . . . 441
Порты . . . 441
Сокеты . . . 443
Протокол UDP и UDP-дейтаграммы . . . 444
Протокол TCP и TCP-сегменты . . . 445
Логические соединения — основа надежности TCP . . . 447
Методы квитирования . . . 451
Метод простоя источника . . . 452
Концепция скользящего окна . . . 453

Передача с возвращением на N пакетов 455
Передача с выборочным повторением. 457
Метод скользящего окна в протоколе TCP 460
Сегменты и поток байтов 460
Система буферов при дуплексной передаче 461
Накопительный принцип квитирования 462
Управление потоком и управление перегрузками 463
Параметры процесса управления потоком 463
Управление перегрузкой 465
Алгоритм «Медленный Старт» 467
Явное уведомление о перегрузке 468
Развитие протокола TCP 469

Глава 16. Протоколы маршрутизации и технология SDN 471
Обзор методов маршрутизации 471
Протокол RIP 476
Построение таблицы маршрутизации 477
Адаптация маршрутизаторов RIP к изменениям состояния сети 480
Пример зацикливания пакетов 481
Методы борьбы с ложными маршрутами в протоколе RIP 483
Протокол OSPF 484
Два этапа построения таблицы маршрутизации 484
Объявления о состоянии связей 485
Метрики 487
Области сети 487
Маршрутизация в неоднородных сетях. Протокол BGP 489
Взаимодействие протоколов маршрутизации 489
Внутренние и внешние шлюзовые протоколы 490
Протокол BGP 492
Групповое вещание 497
Стандартная модель группового вещания IP 497
Адреса группового вещания 498
Протокол IGMP 499
Принципы маршрутизации трафика группового вещания 501
Программно-определяемые сети SDN 503
Недостатки традиционной модели маршрутизации 503
Принципы организации сетей SDN 504
Сети SDN на основе протокола OpenFlow 506
Программируемые протокольно-независимые пакетные процессоры (P4) 508
Виртуализация сетевых функций: NFV 509

Глава 17. IPv6 как развитие стека TCP/IP 513
Исторические предпосылки 513
Система адресации IPv6 514

Отличие от IPv4 . . . 514
Типы адресов IPv6 . . . 515
Индивидуальные адреса . . . 516
Групповые адреса . . . 519
Типичный набор адресов интерфейса IPv6 . . . 521
Формат пакета IPv6 . . . 522
Основной заголовок . . . 523
Дополнительные заголовки . . . 524
Снижение нагрузки на маршрутизаторы . . . 526
Протокол обнаружения соседей Neighbour Discovery . . . 527
Задачи протокола ND и протокол ICMPv6 . . . 527
Сообщения протокола ND . . . 528
Проверка наличия дубликата адреса с помощью протокола ND . . . 530
Разрешение адресов в IPv6 . . . 532
Процесс адаптации версии IPv6 . . . 533
Темпы миграции . . . 533
Проблема интеграции сетей разных технологий . . . 534
Двойной стек, трансляция, туннелирование . . . 535
Способы сосуществования сетей IPv4 и IPv6 . . . 539

Вопросы к части IV . . . 545

ЧАСТЬ V.
ГЛОБАЛЬНЫЕ ПРОВОДНЫЕ КОММУНИКАЦИОННЫЕ СЕТИ

Глава 18. Организация и коммуникационные услуги глобальных сетей . . . 555
Сети операторов связи . . . 555
Услуги операторов связи . . . 556
Потребители услуг . . . 557
Инфраструктура . . . 558
Территория покрытия . . . 559
Взаимоотношения между операторами связи . . . 560
Организация интернета . . . 561

Глава 19. Информационные услуги глобальных сетей . . . 564
Общие принципы организации сетевых служб . . . 565
Веб-служба . . . 567
Веб- и HTML-страницы . . . 567
URL-адрес . . . 568
Веб-клиент и веб-сервер . . . 569
Протокол HTTP . . . 571
Протокол HTTP/3 и транспортный протокол QUIC . . . 575
Формат HTTP-сообщений . . . 578
Динамические и статические веб-страницы . . . 580

Почтовая служба . . . 582
Электронные сообщения . . . 582
Протокол SMTP . . . 584
Непосредственное взаимодействие клиента и сервера . . . 585
Схема с выделенным почтовым сервером . . . 586
Схема с двумя почтовыми серверами-посредниками . . . 588
Протоколы POP3 и IMAP . . . 589
Облачные сервисы . . . 590
Концепция облачных вычислений . . . 590
Определение облачных вычислений . . . 592
Модели облачных сервисов . . . 594
Служба управления сетью . . . 596
Функции систем управления сетью . . . 596
Архитектура систем управления сетью . . . 597
Двухзвенная и трехзвенная схемы управления . . . 598
Взаимодействие менеджера, агента и управляемого объекта . . . 600
Системы управления сетью на основе протокола SNMP . . . 602
Режим удаленного управления и протокол telnet . . . 604
Потоковая телеметрия . . . 605

Глава 20. MPLS — транспортная технология глобальных сетей . . . 607
Принципы работы виртуального канала . . . 607
Эффективность виртуальных каналов . . . 610
Базовые принципы и механизмы MPLS . . . 611
Совмещение коммутации и маршрутизации . . . 612
Пути коммутации по меткам . . . 614
Заголовок MPLS и технологии канального уровня . . . 617
Стек меток . . . 618
Протокол LDP . . . 622
Инжиниринг трафика в MPLS . . . 626
Отказоустойчивость путей в MPLS . . . 634
Общая характеристика . . . 634
Использование иерархии меток для быстрой защиты . . . 636

Глава 21. Виртуальные частные сети . . . 638
Особенности VPN . . . 638
Истинная частная сеть . . . 638
Типы услуг VPN . . . 641
Стандартизация услуг VPN второго уровня . . . 644
Виртуальные частные сети на базе MPLS . . . 646
Технология MPLS VPN второго уровня . . . 646

Вопросы к части V . . . 654

ЧАСТЬ VI.
БЕСПРОВОДНАЯ ПЕРЕДАЧА ДАННЫХ

Глава 22. Особенности беспроводных сетей 661
Ключевые вехи в развитии беспроводных сетей. 661
Нестабильность линий связи 663
Разделяемая среда 664
Зона покрытия и топология 666
Типы узлов в беспроводных сетях 670
Инфраструктурные сети и сети без инфраструктуры 672
Самоорганизующиеся сети AdHoc 673
Мобильность беспроводных узлов 675
О безопасности беспроводных сетей 677
Ограничения энергопотребления 679
Типы беспроводных сетей и радиотехнологий 682

Глава 23. Технологии физического уровня беспроводных сетей 686
Элементы беспроводной линии связи 686
Диапазоны электромагнитного спектра 688
Распространение электромагнитных волн. 690
Типы волн 691
Прямая и непрямая видимость 692
Борьба с искажениями сигнала в беспроводных линиях связи 694
Лицензирование 696
Антенны 696
Изотропные и направленные антенны 697
Прием и передача с использованием нескольких антенн (MIMO) 700
Конфигурации систем с несколькими антеннами 700
Пространственное разнесение 702
Пространственное мультиплексирование 704
Формирование диаграммы направленности и предварительное кодирование 705
Бюджет беспроводной линии связи 706
Техника расширенного спектра 708
Расширение спектра скачкообразной перестройкой частоты FHSS. 708
Прямое последовательное расширение спектра DSSS 711
Множественный доступ с кодовым разделением CDMA. 712
Ортогональное частотное мультиплексирование. 714
Многопользовательский режим OFDMA 717

Глава 24. Сотовые и спутниковые сети 720
Эволюция мобильных сотовых сетей 720
Мобильные сотовые сети 4G/LTE 721
Элементы и структура сети LTE 722
Туннелирование 726

Присоединение устройства пользователя к сети 734
Продвижение пакета через сеть 738
Эстафетная передача и роуминг 739
Спящий режим и пейджинг 743
Радиосеть доступа 744
Передача голоса через сеть LTE 748
Мобильные сотовые сети 5G 760
Новый взгляд на роль мобильных сетей 760
Области применения сетей 5G 761
Виртуализация сети 5G 763
Архитектура сети 5G 764
Новое радио 768
Развитие сетей 5G 772
Спутниковые сети 773
Типы спутниковых сетей 773
Частотные диапазоны 774
Iridium 777
Starlink 778
Прямая связь с телефоном 781

Глава 25. Технологии Wi-Fi и Bluetooth 783
История создания и основные характеристики Wi-Fi 783
Архитектура сети BSS 784
Элементы базовой сети BSS 784
Топология сети BSS 786
Процедура присоединения к сети 788
Стек протоколов 789
Физический уровень BSS 789
MAC-уровень 791
Формат кадров MAC-уровня 791
Доступ к разделяемой среде 794
Предотвращение эффекта скрытой станции 797
Поддержка QoS 799
Агрегирование кадров 800
Централизованный доступ к среде 802
Расширенная архитектура сети Wi-Fi 806
Роуминг в сети ESS 808
Ячеистая архитектура сети Wi-Fi 809
Технология Bluetooth 813
Особенности персональных сетей 813
Архитектура Bluetooth 814
Поиск и стыковка устройств Bluetooth 816
Физический уровень Bluetooth 816

Глава 26. Беспроводные технологии с низким энергопотреблением. 820
Области применения 820
Спектр технологий с низким энергопотреблением 822
Общая платформа 802.15.4 823
Технологии ZigBee, Thread и Wi-SUN FAN 826
Стек протоколов технологии ZigBee 827
Стек протоколов технологий Thread и Wi-SUN FAN 829
Пример сети ZigBee 830
Технология LoRaWAN 831
Назначение и архитектура 831
Управление энергопотреблением 834
Модуляция LoRa 835
Модуляция сигнала LoRa 835

Глава 27. Сети дронов 843
Области применения и классификация дронов 843
Сети с использованием дронов и сети на основе дронов 845
Архитектура связей дронов 847
Схема управления изолированным дроном 847
Централизованная сеть дронов 848
Децентрализованная летающая AdHoc-сеть FANET 850
Архитектуры с несколькими группами дронов 852
Сети FANET 854
Общие свойства и различия сетей MANET, VANET и FANET 854
Навигация 857
Функциональная схема дрона — узла FANET 858
Стек протоколов FANET 861
Уровень сетевых интерфейсов 861
Сетевой уровень стека FANET 868
Прикладной и транспортный уровни 871
Маршрутизация в сетях дронов 872
Особенности маршрутизации в FANET 872
Реактивный протокол AODV 874
Маршрутизация в сетях, устойчивых к разрывам (DTN) 879

Вопросы к части VI 883

ЧАСТЬ VII.
БЕЗОПАСНОСТЬ КОМПЬЮТЕРНЫХ СЕТЕЙ

Глава 28. Основные понятия и принципы информационной безопасности 891
Идентификация, аутентификация и авторизация 891
Модели информационной безопасности 894
Триада «конфиденциальность, доступность, целостность» 894
Гексада Паркера 896

Уязвимость, угроза, атака . . . 897
Ущерб и риск. Управление рисками . . . 900
Типы и примеры атак . . . 901
Пассивные и активные атаки . . . 901
Отказ в обслуживании . . . 902
Внедрение вредоносных программ . . . 904
Кража личности, фишинг . . . 905
Иерархия средств защиты . . . 906
Принципы защиты информационной системы . . . 907
Подход сверху вниз . . . 907
Защита как процесс . . . 908
Эшелонированная защита . . . 908
Сбалансированная защита . . . 910
Компромиссы системы безопасности . . . 911
Шифрование — базовая технология безопасности . . . 912
Основные понятия и определения . . . 912
Симметричное шифрование . . . 913
Проблема распределения ключей . . . 915
Метод Диффи — Хеллмана передачи секретного ключа по незащищенному каналу . . . 916
Концепция асимметричного шифрования . . . 918
Алгоритм асимметричного шифрования RSA . . . 920
Хеш-функции. Односторонние функции шифрования. Проверка целостности . . . 922

Глава 29. Технологии аутентификации, авторизации и управления доступом . . . 924
Технологии аутентификации . . . 924
Факторы аутентификации человека . . . 924
Аутентификация на основе паролей . . . 925
Аутентификация на основе аппаратных аутентификаторов . . . 930
Аутентификация информации. Электронная подпись . . . 933
Аутентификация на основе цифровых сертификатов . . . 935
Аутентификация программных кодов . . . 940
Аутентификация пользователей ОС . . . 941
Технологии управления доступом и авторизации . . . 942
Формы представления ограничений доступа . . . 942
Дискреционный метод управления доступом . . . 945
Мандатный метод управления доступом . . . 946
Ролевое управление доступом . . . 949
Управление доступом в операционных системах . . . 951
Централизованные системы аутентификации и авторизации . . . 951

Глава 30. Технологии безопасности на основе анализа трафика . . . 954
Фильтрация . . . 954
Виды фильтрации . . . 954
Правила фильтрации маршрутизаторов Cisco . . . 955

Файерволы 958
Функциональное назначение файервола 958
Типы файерволов 961
Программные файерволы хоста 964
Влияние DHCP на работу файервола 966
Прокси-серверы 966
Функции прокси-сервера 966
«Проксификация» приложений 969
Трансляция сетевых адресов 969
Традиционная технология NAT 970
Базовая трансляция сетевых адресов 971
Трансляция сетевых адресов и портов 973
Системы мониторинга трафика 975
Анализаторы протоколов 976
Система мониторинга NetFlow 978
Системы обнаружения вторжений 980
Аудит событий безопасности 983
Типовые архитектуры сетей, защищаемых файерволами 985
Логическая сегментация защищаемой сети 985
Архитектура сети с защитой периметра и разделением внутренних зон 987

Глава 31. Атаки на транспортную инфраструктуру сети 990
Атаки на транспортные протоколы 990
TCP-атаки 990
ICMP-атаки 993
UDP-атаки 997
IP-атаки 998
Сетевая разведка 999
Атаки на DNS 1001
DNS-спуфинг 1001
Атаки на корневые DNS-серверы 1002
DDoS-атаки отражением от DNS-серверов 1004
Методы защиты службы DNS 1005
Безопасность маршрутизации на основе BGP 1005
Уязвимости протокола BGP 1005
Инциденты с протоколом BGP 1007
Технологии защищенного канала 1008
Способы образования защищенного канала 1009
Иерархия технологий защищенного канала 1010
Система IPSec 1012

Глава 32. Безопасность программного кода и сетевых служб 1022
Уязвимости программного кода и вредоносные программы 1022
Уязвимости, связанные с нарушением защиты оперативной памяти 1022

Троянские программы 1024
Сетевые черви 1024
Вирусы 1027
Программные закладки 1029
Антивирусные программы 1029
Ботнет 1030
Безопасность веб-сервиса 1031
Безопасность веб-браузера 1032
Приватность и куки 1032
Протокол HTTPS 1033
Безопасность средств создания динамических страниц 1035
Безопасность электронной почты 1036
Угрозы приватности почтового сервиса 1036
Аутентификация отправителя 1037
Шифрование содержимого письма 1040
Защита метаданных пользователя 1040
Спам 1042
Атаки почтовых приложений 1042
Безопасность облачных сервисов 1043
Облачные вычисления как источник угрозы 1043
Облачные сервисы как средство повышения сетевой безопасности 1046

Вопросы к части VII 1049

Ответы 1057
Ответы на вопросы к части I 1057
Ответы к части II 1058
Ответы к части III 1060
Ответы к части IV 1061
Ответы к части V 1063
Ответы к части VI 1064
Ответы к части VII 1066

Рекомендуемая и использованная литература 1069

Список материалов на сайте книги www.olifer.co.uk 1072

Алфавитный указатель 1074

Посвящаем нашим дорогим
Анне, Майклу,
Дане, Полине, Ване и Кате

От авторов

Эта книга является результатом многолетнего опыта преподавания авторами курсов сетевой тематики в аудиториях государственных вузов и различных учебных центров, а также участия в научно-технических разработках, таких как проект Janet, связанный с созданием объединяющей сети кампусов университетов и исследовательских центров Великобритании, и панъевропейские проекты GEANT2, GEANT3 и GEANT4.

Основу книги составили материалы курсов «Проблемы построения корпоративных сетей», «Основы сетевых технологий», «Организация удаленного доступа», «Сети TCP/IP», «Стратегическое планирование сетей масштаба предприятия» и ряда других. Эти материалы прошли успешную проверку в бескомпромиссной и сложной аудитории, состоящей из слушателей с весьма различным уровнем подготовки и кругом профессиональных интересов. Среди них — студенты и аспиранты вузов, сетевые администраторы и интеграторы, начальники отделов автоматизации и преподаватели. Учитывая специфику аудитории, курсы лекций строились так, чтобы начинающий получил основу для дальнейшего изучения, а специалист смог систематизировать и актуализировать имеющиеся знания. В соответствии с такими же принципами написана и эта книга — она является фундаментальным курсом по компьютерным сетям, сочетающим широту охвата основных областей, проблем и технологий этой быстроразвивающейся области знаний с основательным рассмотрением деталей каждой технологии.

Для кого эта книга

Книга предназначена для студентов, аспирантов и технических специалистов, которые хотят получить базовые знания о принципах построения компьютерных сетей, понять особенности традиционных и перспективных технологий локальных и глобальных сетей, изучить способы создания крупных составных сетей и управления такими сетями.

Книга будет полезна начинающим специалистам в области сетевых технологий, имеющим только общие представления о работе сетей из опыта общения с персональными компьютерами и интернетом, но стремящимся получить фундаментальные знания, позволяющие продолжить изучение сетей самостоятельно.

Сложившимся сетевым специалистам книга может помочь в знакомстве с теми технологиями, с которыми им не приходилось сталкиваться в практической работе, систематизировать имеющиеся знания, стать справочником, позволяющим найти описание конкретного протокола, формата кадра и т. п. Кроме того, книга дает необходимую теоретическую основу для подготовки к сертификационным экзаменам таких компаний, как Cisco и Juniper.

Студенты организаций высшего профессионального образования, обучающиеся по направлению «09.00.00. Информатика и вычислительная техника» и по специальностям «Вычислительные машины, комплексы, системы и сети», «Автоматизированные машины, комплексы, системы и сети», «Программное обеспечение вычислительной техники и автоматизированных систем», могут использовать книгу в качестве учебного пособия.

Из предисловия к юбилейному шестому изданию

Это издание в некотором смысле особенное — прошло ровно 20 лет с момента выхода книги в свет. Двадцать лет — это немалый срок, за это время дети наших первых читателей подросли и, возможно, стали интересоваться компьютерными сетями. И, возможно, у них в руках окажется 6-е издание книги «Компьютерные сети. Принципы, технологии, протоколы». Эта книга значительно отличается от той книги, которую читали их родители. Многое из того, что так интересовало читателей конца 90-х годов, — например, правило четырех хабов, согласование сетей IP и IPX или сравнение технологий 100VG-AnyLAN и FDDI — совсем не упоминается в последних изданиях. За 20 лет немало технологий прошли полный цикл от модного термина и всеобщего признания к практически полному забвению. Каждое новое издание книги в той или иной мере отражало изменения ландшафта сетевых технологий.

Изменения в седьмом издании

Единственным способом существования долгоживущих учебников в быстроизменяющейся области компьютерных сетей является их регулярное и существенное обновление. В седьмое издание, так же, как и во все предыдущие, внесено много изменений, примерно четверть книги написана заново. Для того чтобы книга не вышла за пределы отведенного ей объема, а также из-за снижения актуальности тех или иных технологий на сайт книги www.olifer.co.uk были перенесены материалы об агрегировании линий связи в локальных сетях, описание технологий первичных сетей PDH/ SDH и сетей доступа ADSL и PON, а также некоторые другие. Для ссылки на материалы, помещенные на сайт, используется значок **(*S*)** в соответствующих местах книги.

В то же время в книге появилось много нового: описание технологий Ethernet 800G и 1,6T, материал о версиях HTTP с учетом использования нового транспортного протокола QUIK, раздел об управлении перегрузками в протоколе TCP, расширенная классификация протоколов маршрутизации и описание протоколов OSPF и BGP, сегментная маршрутизация в MPLS и инжиниринг трафика на ее основе, более подробное описание виртуальных частных сетей.

Коренной переработке подверглась часть книги, посвященная беспроводной передаче данных. Эта часть книги значительно увеличилась в объеме и теперь состоит из шести глав вместо трех. Первая глава части посвящена рассмотрению проблем, с которыми мы раньше не сталкивались в этой книге и которые характерны именно для беспроводных сетей: мобильность как норма, нестабильность линий связи, быстрые изменения топологии сети, учет энергозатрат при разработке протоколов всех уровней.

В новом издании более глубоко и детально рассматриваются технологии мобильных сотовых сетей и Wi-Fi. Добавлен раздел, посвященный спутниковой связи в системах Iridium и Starlink.

При рассмотрении технологий физического уровня беспроводных сетей добавлено описание технологии мультиплексирования OFDMA, на основе которой организуется многопользовательский режим доступа в мобильных сотовых сетях 4G/5G и в новых версиях протоколов Wi-Fi.

Появился раздел, посвященный быстроразвивающимся беспроводным технологиям интернета вещей, характеризующимся низким энергопотреблением: ZigBee, Thread, Wi-SUN FAN и LoRaWAN.

Тему беспроводных сетей передачи данных завершает глава 27 «Сети дронов», которую можно рассматривать как краткое, но многостороннее введение в область коммуникаций в рое дронов. В этой главе рассматриваются области применения, классификация и архитектура компьютерных сетей нового типа, в которых узлами сети являются БПЛА. Приводится описание технологий радиосвязи, учитывающих специфику дронов.

Вопросы к разделам книги также были переработаны.

Благодарности

Мы благодарим наших читателей за их многочисленные пожелания, вопросы и замечания. Мы признательны также всем сотрудникам издательства «Питер», которые принимали участие в создании этой книги. Особая благодарность президенту издательства «Питер» Вадиму Усманову, руководителю редакции Юлии Сергиенко, ведущему редактору Екатерине Строгановой и литературному редактору Юлии Широниной.

Мы с благодарностью примем отзывы читателей по адресам

victor@olifer.co.uk и natalia@olifer.co.uk.

Виктор Олифер

Наталья Олифер

От издательства

Ваши замечания, предложения, вопросы отправляйте по адресу электронной почты comp@piter.com (издательство «Питер», компьютерная редакция).

Мы будем рады узнать ваше мнение!

На веб-сайте издательства http://www.piter.com вы найдете подробную информацию о наших книгах.

Часть I

Основы сетей передачи данных

- ❑ Глава 1. Эволюция компьютерных сетей
- ❑ Глава 2. Общие принципы построения сетей
- ❑ Глава 3. Коммутация каналов и пакетов
- ❑ Глава 4. Стандартизация и классификация сетей
- ❑ Глава 5. Сетевые характеристики и качество обслуживания

Процесс познания всегда развивается по спирали. Мы не можем сразу понять и осознать сложное явление, мы должны рассматривать его с разных точек зрения, в целом и по частям, изолированно и во взаимодействии с другими явлениями, накапливая знания постепенно, время от времени возвращаясь к уже, казалось бы, понятому и с каждым новым витком все больше проникая в суть явления. Хорошим подходом является первоначальное изучение общих принципов некоторой области знаний с последующим детальным рассмотрением реализации этих принципов в конкретных методах, технологиях или конструкциях. Первая часть книги и является таким «первым витком» изучения компьютерных сетей.

Изучение общих принципов построения компьютерных сетей поможет вам в дальнейшем быстрее разбираться с любой конкретной сетевой технологией. Однако известное высказывание «Знание нескольких принципов освобождает от запоминания множества фактов» не стоит воспринимать буквально — хороший специалист, конечно же, должен знать множество деталей и фактов. Знание принципов позволяет систематизировать эти частные сведения, связать их друг с другом в стройную систему и тем самым использовать более осознанно и эффективно. Конечно, изучение принципов перед изучением конкретных технологий — задача непростая, особенно для читателей с практическим складом ума. Кроме того, всегда есть опасность неверного понимания какого-нибудь общего утверждения без проверки его в практической реализации. Поэтому мы просим читателей поверить нам пока на слово, что игра стоит свеч, а также последовать совету: в ходе изучения материала последующих глав книги время от времени возвращайтесь к теоретическим вопросам и проверяйте себя, так ли вы понимали те или иные механизмы, когда изучали их впервые.

Часть, а вместе с ней и книга, открывается главой об эволюции компьютерных сетей. История любой отрасли науки и техники позволяет не только удовлетворить естественное любопытство, но и глубже понять сущность основных достижений в этой отрасли, осознать существующие тенденции и оценить перспективность тех или иных направлений развития,

В следующих двух главах рассматриваются фундаментальные концепции компьютерных сетей — коммутация, маршрутизация, мультиплексирование, адресация. Изучаются методы продвижения

пакетов — дейтаграммная передача, передача с установлением логического соединения и техника виртуальных каналов.

Важной темой данной части книги является рассматриваемая в четвертой главе стандартизация архитектуры компьютерной сети, идеологической основой которой служит модель взаимодействия открытых систем (OSI).

Последняя глава этой части книги посвящена сетевым характеристикам и проблемам качества обслуживания. Новая роль компьютерных сетей как основы для создания следующего поколения публичных сетей, предоставляющих все виды информационных услуг и переносящихданные, а также аудио- и видеотрафик, привела к проникновению методов обеспечения качества обслуживания практически во все коммуникационные технологии. Таким образом, концепции качества обслуживания, которые достаточно долго рассматривались как вспомогательное направление сетевой отрасли, вошли в число базовых принципов построения компьютерных сетей.

ГЛАВА 1 Эволюция компьютерных сетей

Два корня компьютерных сетей

Компьютерные сети, которым посвящена данная книга, отнюдь не являются единственным видом сетей, созданным человеческой цивилизацией. Даже водопроводы Древнего Рима можно рассматривать как один из наиболее древних примеров сетей, покрывающих большие территории и обслуживающих многочисленных клиентов. Другой, менее экзотический пример — электрические сети. В них легко найти аналоги компонентов любой территориальной компьютерной сети: источникам информационных ресурсов соответствуют электростанции, магистралям — высоковольтные линии электропередачи, сетям доступа — трансформаторные подстанции, клиентским терминалам — осветительные и бытовые электроприборы.

Компьютерные сети, называемые также **сетями передачи данных**, являются логическим результатом эволюции двух важнейших научно-технических отраслей современной цивилизации — вычислительной техники и телекоммуникационных технологий.

С одной стороны, компьютерные сети представляют собой группу компьютеров, согласованно решающих набор взаимосвязанных задач, обмениваясь данными в автоматическом режиме. С другой стороны, компьютерные сети могут рассматриваться как средство передачи информации на большие расстояния, для чего в них применяются методы кодирования и мультиплексирования данных, получившие развитие в различных телекоммуникационных системах, например телефонных сетях (рис. 1.1).

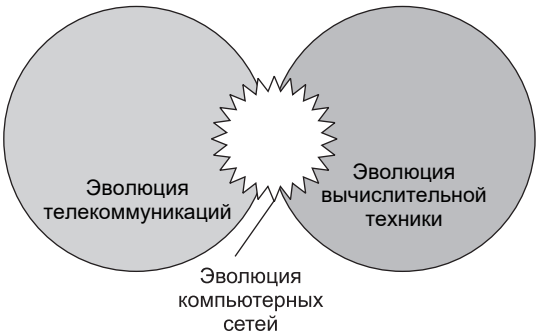


Рис. 1.1. Эволюция компьютерных сетей на стыке вычислительной техники и телекоммуникационных технологий

Первые компьютерные сети

Системы пакетной обработки

Обратимся сначала к компьютерному корню вычислительных сетей. Первые компьютеры 1950-х годов — большие, громоздкие и дорогие — предназначались для очень небольшого числа избранных пользователей. Часто эти монстры занимали целые здания. Такие компьютеры не были предназначены для интерактивной работы пользователя, а применялись в режиме пакетной обработки.

Системы пакетной обработки, как правило, строились на базе мейнфрейма — мощного и надежного компьютера универсального назначения. Пользователи подготавливали перфокарты, содержащие данные и команды программ, и передавали их в вычислительный центр (рис. 1.2). Задания нескольких пользователей группировались в пакет, который принимался на выполнение. Оператор мейнфрейма вводил карты пакета в компьютер, который обрабатывал задания в многопрограммном режиме, оптимизируя распределение процессора и устройств ввода-вывода между заданиями для достижения максимальной производительности вычислений. Распечатанные результаты пользователи получали обычно только на следующий день. Таким образом, одна неверно набитая карта означала как минимум суточную задержку. Конечно, для пользователей интерактивный режим работы, при котором можно с терминала оперативно руководить процессом обработки своих данных, был бы удобнее. Но интересами пользователей на первых этапах развития вычислительных систем в значительной степени пренебрегали. Во главу угла ставилась эффективность работы самого дорогого устройства вычислительной машины — процессора, даже в ущерб эффективности работы использующих его специалистов.



Рис. 1.2. Централизованная система на базе мейнфрейма

Многотерминальные системы — прообраз сети

По мере удешевления процессоров в начале 60-х годов появились новые способы организации вычислительного процесса, которые позволили учесть интересы пользователей. Начали развиваться интерактивные **многотерминальные системы разделения времени** (рис. 1.3). В таких системах каждый пользователь получал собственный терминал, с помощью которого он мог вести диалог с компьютером. Количество одновременно работающих с компьютером пользователей определялось его мощностью: время реакции вычислительной системы должно было быть достаточно мало, чтобы пользователю была не слишком заметна параллельная работа с компьютером других пользователей.

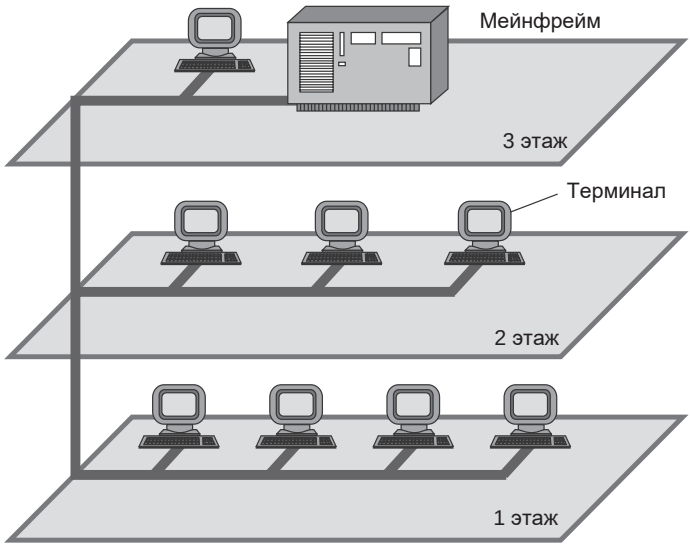


Рис. 1.3. Многотерминальная система — прообраз вычислительной сети

Терминалы, выйдя за пределы вычислительного центра, рассредоточились по всему предприятию. И хотя вычислительная мощность оставалась полностью централизованной, некоторые функции — такие как ввод и вывод данных — стали распределенными. Подобные многотерминальные централизованные системы внешне уже были очень похожи на локальные вычислительные сети. Действительно, рядовой пользователь работу за терминалом мейнфрейма воспринимал примерно так же, как сейчас он воспринимает работу за подключенным к сети персональным компьютером. Пользователь мог получить доступ к общим файлам и периферийным устройствам, при этом у него поддерживалась полная иллюзия единоличного владения компьютером, так как он мог запустить нужную ему программу в любой момент и почти сразу же получить результат (некоторые далекие от вычислительной техники пользователи даже были уверены, что все вычисления выполняются внутри их дисплея).

Многотерминальные системы, работающие в режиме разделения времени, стали прообразом локальных вычислительных сетей.

Однако до появления локальных сетей нужно было пройти еще большой путь, так как многотерминальные системы, хотя и имели внешние черты распределенных систем, все еще поддерживали централизованную обработку данных.

К тому же потребность предприятий в создании локальных сетей в это время еще не созрела — в одном здании просто нечего было объединять в сеть, так как из-за высокой стоимости вычислительной техники предприятия не могли себе позволить роскошь приобретения нескольких компьютеров. В этот период был справедлив так называемый **закон Гроша**, который эмпирически отражал достигнутый уровень технологии. В соответствии с этим законом *производительность компьютера была пропорциональна квадрату его стоимости*. Отсюда следовало, что за одну и ту же сумму было выгоднее купить одну мощную машину, чем две менее мощных — их суммарная мощность оказывалась намного ниже мощности дорогой машины.

Первые глобальные сети

А вот потребность в соединении нескольких компьютеров, находящихся на большом расстоянии друг от друга, к этому времени уже вполне назрела. Началось все с решения более простой задачи — доступа к отдельному компьютеру с терминалов, удаленных от него на многие сотни, а то и тысячи километров. Терминалы соединялись с компьютером через телефонные сети с помощью модемов, позволив многочисленным пользователям получать удаленный доступ к разделяемым ресурсам мощных суперкомпьютеров. Затем появились системы, в которых наряду с удаленными соединениями типа *терминал — компьютер* были реализованы и удаленные связи типа *компьютер — компьютер*.

Разнесенные территориально компьютеры получили возможность *обмениваться данными в автоматическом режиме*, что, собственно, и является базовым признаком любой вычислительной сети.

На основе подобного механизма в первых сетях были реализованы службы обмена файлами, синхронизации баз данных, электронной почты и другие, ставшие теперь традиционными сетевые службы.

Итак, хронологически первыми появились **глобальные сети** (Wide Area Network, WAN), то есть сети, объединяющие территориально рассредоточенные компьютеры, возможно, находящиеся в различных городах и странах.

Именно при построении глобальных сетей были впервые предложены и отработаны многие основные идеи, лежащие в основе современных вычислительных сетей. Такие, например, как многоуровневое построение коммуникационных протоколов, концепции коммутации и маршрутизации пакетов.

Глобальные компьютерные сети очень многое унаследовали от других, гораздо более старых и распространенных глобальных сетей — *телефонных*. Главное технологическое новшество, которое привнесли с собой первые глобальные компьютерные сети, состояло в отказе от **принципа коммутации каналов**, на протяжении многих десятков лет успешно использовавшегося в телефонных сетях.

Выделяемый на все время сеанса связи составной телефонный канал, передающий информацию с постоянной скоростью, не мог эффективно использоваться пульсирующим трафиком компьютерных данных, у которого периоды интенсивного обмена чередуются с продолжи-

тельными паузами. Натурные эксперименты и математическое моделирование показали, что пульсирующий и в значительной степени не чувствительный к задержкам компьютерный трафик гораздо эффективнее передается сетями, работающими по **принципу коммутации пакетов**, когда данные разделяются на небольшие порции — пакеты, которые самостоятельно перемещаются по сети благодаря наличию адреса конечного узла в заголовке пакета.

Так как прокладка высококачественных линий связи на большие расстояния обходится очень дорого, то в первых глобальных сетях часто использовались уже существующие линии связи, изначально предназначенные совсем для других целей. Например, в течение многих лет глобальные сети строились на основе телефонных каналов тональной частоты, способных в каждый момент времени вести передачу только одного разговора в аналоговой форме. Поскольку скорость передачи дискретных компьютерных данных по таким каналам была очень низкой (десятки килобит в секунду), набор предоставляемых услуг в глобальных сетях подобного типа обычно ограничивался передачей файлов (преимущественно в фоновом режиме) и электронной почтой. Помимо низкой скорости такие каналы имеют и другой недостаток — они вносят значительные искажения в передаваемые сигналы. Поэтому протоколы глобальных сетей, построенных с использованием каналов связи низкого качества, отличались сложными процедурами контроля и восстановления данных. Типичным примером таких сетей являются сети X.25, разработанные еще в начале 70-х годов.

ПРИМЕЧАНИЕ

При написании этой главы авторы столкнулись с дилеммой: невозможно рассказывать об истории отрасли, не называя конкретные технологии и концепции. Но в то же время невозможно давать пояснения этих технологий и концепций, так как читатель, перелистывающий первые страницы, еще не готов к восприятию объяснений. Авторы пошли по пути компромисса, отложив на будущее исчерпывающие пояснения многих терминов ради того, чтобы в самом начале изучения компьютерных сетей читатель имел возможность представить картину эволюции компьютерных сетей во всем ее красочном многообразии.

В 1969 году министерство обороны США инициировало работы по объединению в единую сеть суперкомпьютеров оборонных и научно-исследовательских центров. Эта сеть, получившая название ARPANET, стала отправной точкой для создания первой и самой известной ныне глобальной сети мирового масштаба — **Internet**.

Сеть ARPANET объединяла компьютеры разных типов, работавшие под управлением различных операционных систем (ОС) с дополнительными модулями, реализующими коммуникационные протоколы, общие для всех компьютеров сети. ОС этих компьютеров можно считать *первыми* **сетевыми операционными системами**.

Сетевые ОС позволили не только рассредоточить пользователей между несколькими компьютерами (как в многотерминальных системах), но и организовать распределенные хранение и обработку данных. Любая сетевая операционная система, с одной стороны, выполняет все функции локальной операционной системы, а с другой стороны, обладает некоторыми дополнительными средствами, позволяющими ей взаимодействовать через сеть с операционными системами других компьютеров. Программные модули, реализующие сетевые функции, появлялись в операционных системах постепенно, по мере развития сетевых технологий, аппаратной базы компьютеров и возникновения новых задач, требующих сетевой обработки.

Прогресс глобальных компьютерных сетей во многом определялся прогрессом телефонных сетей. Так появившаяся в начале 60-х годов в телефонных сетях передача голоса в цифровой форме привела к тому, что высокоскоростные цифровые каналы, соединяющие автоматические телефонные станции, стали использоваться и в глобальных компьютерных сетях.

Первые локальные сети

Важное событие, повлиявшее на эволюцию компьютерных сетей, произошло в начале 70-х годов. В результате технологического прорыва в области производства компьютерных компонентов появились **большие интегральные схемы** (БИС). Их сравнительно невысокая стоимость и хорошие функциональные возможности привели к созданию **мини-компьютеров**, которые стали реальными конкурентами мейнфреймов. Эмпирический закон Гроша перестал соответствовать действительности, так как десяток мини-компьютеров, имея ту же стоимость, что и один мейнфрейм, решали некоторые задачи (как правило, хорошо распараллеливаемые) быстрее.

Даже небольшие подразделения предприятий получили возможность иметь собственные компьютеры. Мини-компьютеры решали задачи управления технологическим оборудованием, складом и другие задачи уровня отдела предприятия. Таким образом, появилась концепция распределения компьютерных ресурсов по всему предприятию. Однако при этом все компьютеры одной организации по-прежнему продолжали работать *автономно* (рис. 1.4).

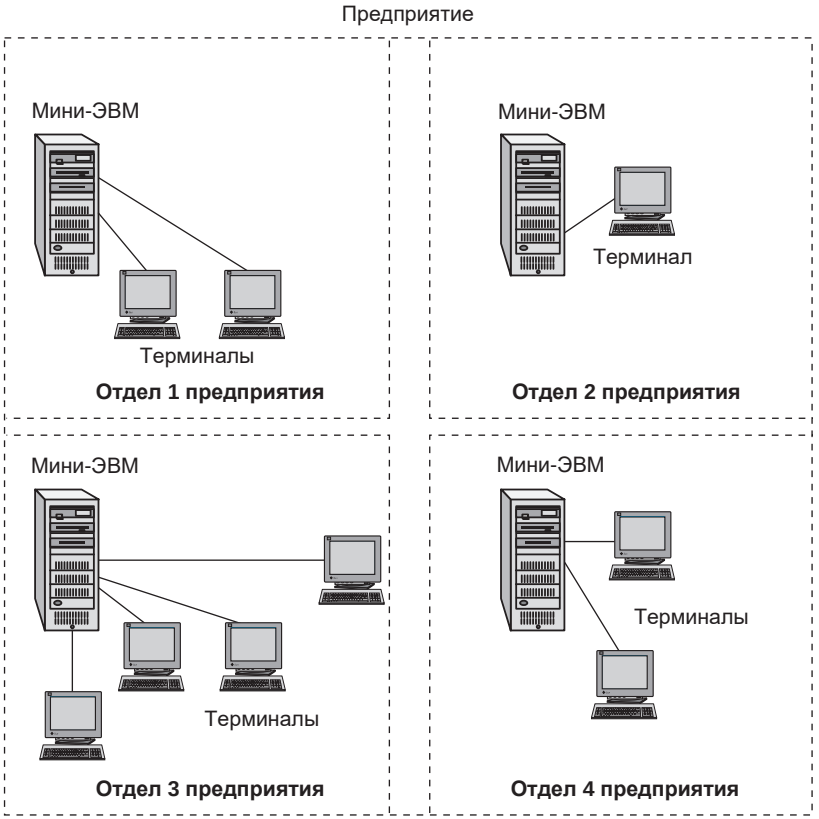


Рис. 1.4. Автономное использование нескольких мини-компьютеров на одном предприятии

Шло время, потребности пользователей вычислительной техники росли. Их уже не удовлетворяла изолированная работа на собственном компьютере, им хотелось в автоматическом режиме обмениваться компьютерными данными с пользователями других подразделений. Ответом на эту потребность и стало появление первых локальных вычислительных сетей (рис. 1.5).

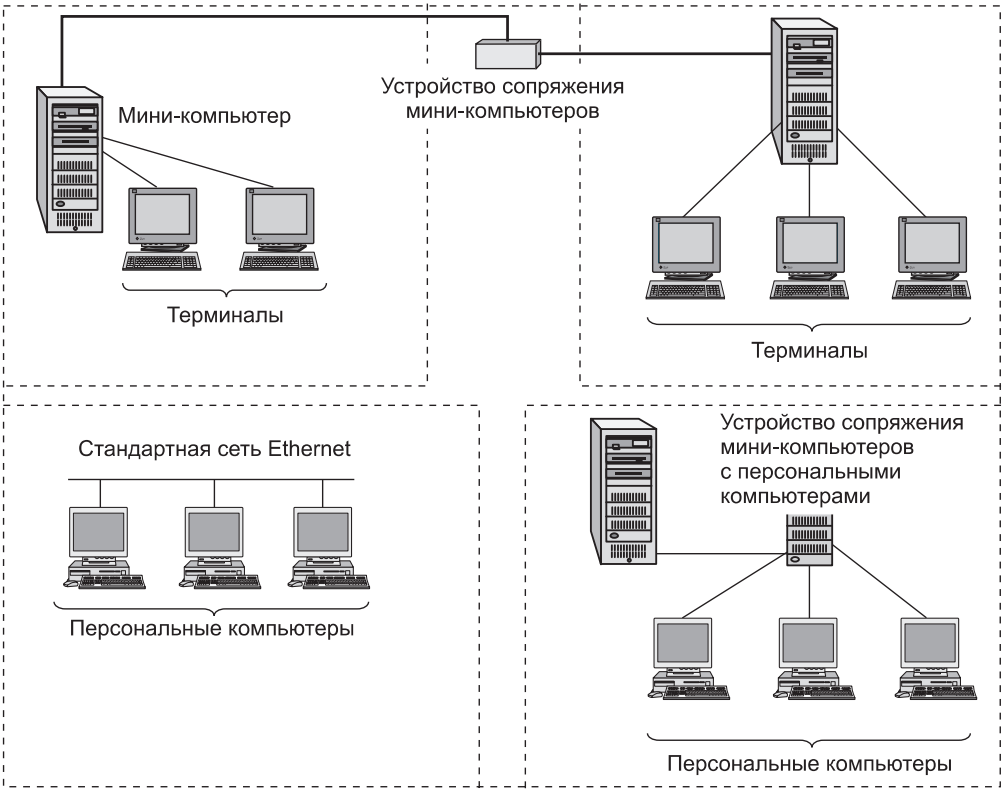


Рис. 1.5. Различные типы связей в первых локальных сетях

Локальные сети (Local Area Network, LAN) — это объединения компьютеров, сосредоточенных на небольшой территории, обычно в радиусе не более 1–2 км, хотя в отдельных случаях локальная сеть может иметь и большие размеры, например, несколько десятков километров. Обычно локальная сеть представляет собой коммуникационную систему, принадлежащую одной организации.

На первых порах для соединения компьютеров друг с другом использовались *нестандартные* сетевые технологии.

Сетевая технология — это согласованный набор программных и аппаратных средств (например, драйверов, сетевых адаптеров, кабелей и разъемов), а также механизмов передачи данных по линиям связи, достаточный для построения вычислительной сети.

Разнообразные устройства сопряжения, использующие собственные способы представления данных на линиях связи, свои типы кабелей и т. п., могли соединять только те конкретные модели компьютеров, для которых были разработаны, например, мини-компьютеры PDP-11 с мейнфреймом IBM 360 или мини-компьютеры HP с микрокомпьютерами LSI-11. Такая ситуация создала большой простор для творчества студентов — названия многих курсовых и дипломных проектов начинались тогда со слов «Устройство сопряжения...».

В середине 80-х годов положение дел в локальных сетях кардинально изменилось. Утвердились *стандартные сетевые технологии* объединения компьютеров в сеть — Ethernet, Arcnet, Token Ring, Token Bus, несколько позже — FDDI.

Мощным стимулом для их появления послужили **персональные компьютеры**. Эти массовые продукты стали идеальными элементами построения сетей — с одной стороны, они были достаточно мощными, чтобы обеспечивать работу сетевого программного обеспечения, а с другой — явно нуждались в объединении своей вычислительной мощности для решения сложных задач, а также разделения дорогих периферийных устройств и дисковых массивов. Поэтому персональные компьютеры стали преобладать в локальных сетях, причем не только в качестве клиентских компьютеров, но и в качестве центров хранения и обработки данных, то есть сетевых серверов, потеснив с этих привычных ролей миникомпьютеры и мейнфреймы.

Все стандартные технологии локальных сетей опирались на тот же принцип коммутации, который был с успехом опробован и доказал свои преимущества при передаче трафика данных в глобальных компьютерных сетях — *принцип коммутации пакетов*.

Стандартные сетевые технологии превратили процесс построения локальной сети из решения нетривиальной технической проблемы в рутинную работу. Для создания сети достаточно было приобрести стандартный кабель, сетевые адаптеры соответствующего стандарта, например Ethernet, вставить адаптеры в компьютеры, присоединить их к кабелю стандартными разъемами и установить на компьютеры одну из популярных сетевых операционных систем, например Novell NetWare.

Разработчики локальных сетей привнесли много нового в организацию работы пользователей. Так, стало намного проще и удобнее, чем в глобальных сетях, получать доступ к общим сетевым ресурсам. Последствием и одновременно движущей силой такого прогресса стало появление огромного числа непрофессиональных пользователей, освобожденных от необходимости изучать специальные (и достаточно сложные) команды для сетевой работы.

Конец 90-х выявил явного лидера среди технологий локальных сетей — семейство *Ethernet*, в которое вошли классическая технология Ethernet со скоростью передачи 10 Мбит/с, а также Fast Ethernet со скоростью 100 Мбит/с и Gigabit Ethernet со скоростью 1000 Мбит/с.

Простые алгоритмы работы этой технологии предопределяют низкую стоимость оборудования Ethernet. Широкий диапазон иерархии скоростей позволяет рационально строить локальную сеть, выбирая ту технологию семейства, которая в наибольшей степени отвечает задачам предприятия и потребностям пользователей.

Конвергенция сетей

Конвергенция локальных и глобальных сетей

В конце 80-х годов отличия между локальными и глобальными сетями проявлялись весьма отчетливо.

- *Протяженность и качество линий связи.* Локальные компьютерные сети по определению отличаются от глобальных сетей небольшими расстояниями между узлами сети. Это в принципе делает возможным использование в локальных сетях более дорогих качественных линий связи. В глобальных сетях 80-х годов преобладали низкоскоростные телефонные линии связи, передающие дискретную информацию компьютеров со сравнительно частыми искажениями.
- *Сложность методов передачи данных.* В условиях низкой надежности физических каналов в глобальных сетях требуются более сложные, чем в локальных сетях, методы передачи данных и соответствующее оборудование.
- *Скорость обмена данными* в локальных сетях (10, 16 и 100 Мбит/с) в то время была существенно выше, чем в глобальных (от 2,4 Кбит/с до 2 Мбит/с).
- *Разнообразие услуг.* Высокие скорости обмена данными позволили предоставлять в локальных сетях широкий спектр услуг — это прежде всего разнообразные способы совместного использования файлов, устройств печати, модемов, факсов, доступ к единой базе данных, электронная почта и др. В то же время глобальные сети в основном ограничивались почтовыми и файловыми услугами в их простейшем (не самом удобном для пользователя) виде.

Постепенно различия между локальными и глобальными сетевыми технологиями стали сглаживаться. Изолированные ранее локальные сети начали объединять друг с другом, при этом в качестве связующей среды использовались глобальные сети. Интеграция локальных и глобальных сетей привела к значительному взаимопроникновению соответствующих технологий.

Сближение в методах передачи данных произошло за счет использования одной и той же среды передачи данных — *оптического волокна* — и одних и тех же принципов кодирования передаваемых данных — цифрового (дискретного) кодирования. Оптоволоконные кабели стали использоваться практически во всех технологиях локальных сетей для скоростного обмена информацией на расстояниях свыше 100 метров, на них же стали строиться магистрали первичных сетей SDH и DWDM, предоставляющих свои цифровые каналы для объединения оборудования глобальных компьютерных сетей.

Высокое качество цифровых каналов изменило требования к протоколам глобальных сетей. На первый план вместо процедур обеспечения надежности вышли процедуры обеспечения гарантированной средней скорости доставки информации пользователям, а также механизмы приоритетной обработки пакетов особенно чувствительного к задержкам трафика, например голосового. Эти изменения нашли отражение в таких технологиях глобальных сетей 90-х годов, как Frame Relay и ATM. В этих технологиях предполагается, что искажение битов происходит настолько редко, что ошибочный пакет выгоднее просто уничтожить, а все проблемы, связанные с его потерей, перепоручить программному обеспечению более высокого уровня, которое непосредственно не входит в состав сетей Frame Relay и ATM.

Огромный вклад в сближение локальных и глобальных сетей внесло доминирование *протокола IP*. Этот протокол может работать поверх любых технологий локальных и глобальных сетей (Ethernet, MPLS, Token Ring, ATM, Frame Relay), объединяя различные подсети в единую составную сеть.

Начиная с 90-х годов компьютерные глобальные сети, работающие на основе скоростных цифровых каналов, существенно расширили спектр предоставляемых услуг и догнали в этом отношении локальные сети. Стало возможным создание служб, работа которых связана с доставкой пользователю больших объемов информации в реальном времени — изображений, видеофильмов, голоса, в общем, всего того, что получило название мультимедийной информации. Наиболее яркий пример — гипертекстовая информационная служба World Wide Web (веб-служба), ставшая основным поставщиком информации в Интернете. Ее интерактивные возможности превзошли возможности многих аналогичных служб локальных сетей, так что разработчикам приложений локальных сетей пришлось просто позаимствовать эту службу у глобальных сетей. Процесс переноса технологий из глобальной сети Интернет в локальные приобрел такой массовый характер, что появился даже специальный термин — **intranet-технологии** (intra — внутренний).

Истинно универсальной сетевой технологией стала технология Ethernet. Долгие годы Ethernet была технологией только локальных сетей, однако, дополненная новыми функциями и новыми уровнями скоростей, она (называемая в этом варианте Carrier Ethernet, то есть Ethernet операторского класса) сегодня преобладает и на линиях связи глобальных сетей. Следствием доминирования технологии Ethernet в первом десятилетии XXI века стало упрощение структуры как локальных, так и глобальных сетей - в подавляющем большинстве подсетей сегодня работает протокол Ethernet, а объединяются подсети в со ставную сеть с помощью протокола IP.

Еще одним признаком сближения локальных и глобальных сетей является появление сетей, занимающих промежуточное положение между локальными и глобальными сетями. **Городские сети**, или **сети мегаполисов** (Metropolitan Area Network, MAN), предназначены для обслуживания территории крупного города.

Эти сети используют цифровые линии связи, часто оптоволоконные, со скоростями на магистрали 10 Гбит/с и выше. Они обеспечивают экономичное соединение локальных сетей между собой, а также выход в глобальные сети. Сети MAN первоначально были разработаны только для передачи данных, но сейчас перечень предоставляемых ими услуг расширился, в частности, они поддерживают видеоконференции и интегральную передачу голоса и текста. Современные сети MAN отличаются разнообразием предоставляемых услуг, позволяя своим клиентам объединять коммуникационное оборудование различного типа, в том числе офисные АТС.

Конвергенция компьютерных и телекоммуникационных сетей

Начиная с 1980-х годов предпринимаются попытки создания универсальной, так называемой **мультисервисной сети**, способной предоставлять услуги как компьютерных, так и телекоммуникационных сетей.

К телекоммуникационным сетям относятся радиосети, телефонные и телевизионные сети. Главное, что объединяет их с компьютерными сетями, — это то, что в качестве ресурса, предоставляемого клиентам, выступает информация. Однако имеется некоторая специфика. Так, изначально компьютерные сети разрабатывались для передачи алфавитно-цифровой информации, которую часто называют просто *данными*, поэтому у компьютерных сетей имеется и другое название — **сети передачи данных**, в то время как **телекоммуникационные** сети были созданы для передачи *голосовой и видеоинформации*.

Сегодня мы являемся свидетелями конвергенции телекоммуникационных и компьютерных сетей, которая идет по нескольким направлениям.

Прежде всего наблюдается *сближение видов услуг*, предоставляемых клиентам. Первая попытка создания мультисервисной сети, способной оказывать различные услуги, в том числе услуги телефонии и передачи данных, привела к появлению в 80-х годах технологии **цифровых сетей с интегрированным обслуживанием** (Integrated Services Digital Network, ISDN). Но на практике сети ISDN так и остались телефонными сетями, а роль глобальной **мультисервисной сети нового поколения** стал играть **интернет**.

Интернет превратился из сети, предназначенной для оказания небольшого набора услуг передачи данных, основными из которых были передача файлов и обмен текстовыми почтовыми сообщениями, в действительно мультисервисную сеть. Интернет может оказывать все виды телекоммуникационных услуг, в том числе услуг мгновенных сообщений, видео и аудиоконференций, IP-телефонии, IP-телевидения, а также услуг многочисленных социальных сетей. Очевидно, что мультисервисность сети интернет в будущем будет только возрастать.

Прорывом в процессе конвергенции сетей явилось появление **смартфонов** — терминальных устройств, которые объединили в себе функции мобильных телефонов и персональных компьютеров. Для поддержки таких новых функций телефона современная *мобильная телефонная сеть* также стала мультисервисной сетью — она предоставляет полный набор как телефонных, так и компьютеризованных информационных услуг (просмотр веб-страниц в такой же удобной форме, как и на экране компьютера, услуги электронной почты и видеоконференций, просмотр фильмов, публикация информации в социальных сетях и т. п.).

Технологическое сближение сетей происходит по нескольким направлениям. Прежде всего это использование *цифровой передачи* для разных типов информации. Представление голоса и изображения в цифровой форме сделало принципиально возможной передачу телефонного, видео и компьютерного трафиков по одним и тем же цифровым каналам.

Важным шагом телефонии навстречу компьютерным сетям стало использование наряду с изначально присущей им коммутацией каналов *методов коммутации пакетов*. Так, для передачи служебных сообщений (называемых сообщениями сигнализации) применяются протоколы коммутации пакетов, аналогичные протоколам компьютерных сетей, а для передачи собственно голоса между абонентами коммутируется традиционный составной канал.

Однако ситуация изменилась в пользу пакетных методов коммутации даже при передаче голоса. У этой тенденции есть достаточно очевидная причина — на основе метода коммутации пакетов можно более эффективно использовать пропускную способность каналов связи и коммутационного оборудования. Например, паузы в телефонном разговоре могут составлять до 40 % общего времени соединения, однако только пакетная коммутация позволяет «вырезать» паузы и использовать высвободившуюся пропускную способность канала для передачи трафика других абонентов.

Обращение к технологии коммутации пакетов для одновременной передачи через пакетные сети разнородного трафика — голоса, видео и текста — сделало актуальной разработку новых методов обеспечения требуемого **качества обслуживания** (Quality of Service, QoS). Методы QoS призваны минимизировать уровень задержек для чувствительного к ним трафика, например голосового, и одновременно гарантировать среднюю скорость и динамичную передачу пульсаций для трафика данных.

Однако неверно было бы говорить, что методы коммутации каналов морально устарели и у них нет будущего. На новом витке спирали развития они находят свое применение, но уже в новых технологиях, таких как технологии **первичных сетей**, служащих основой как для компьютерных, так и для телефонных сетей: Optical Transport Networks (OTN) и Dense Wavelength Division Multiplexing (DWDM).

Еще одним проявлением технологической конвергенции стало использование в телекоммуникационных сетях традиционного для компьютерных сетей *протокола IP*. Например, этот протокол нашел свое применение в технологиях мобильных телефонных сетей 3-го, 4-го и 5-го поколений (3G, 4G, 5G).

Компьютерные сети также многое позаимствовали у телефонных и телевизионных сетей. В частности, они взяли на вооружение методы обеспечения отказоустойчивости телефонных сетей, за счет которых последние демонстрируют высокую степень надежности, так недостающей порой интернету и корпоративным сетям.

Учитывая описанный процесс эволюции, мы в этой книге будем использовать устоявшийся термин «телекоммуникационная сеть» в расширенном понимании — то есть подразумевая под телекоммуникационными сетями и компьютерные сети.

Интернет как фактор развития сетевых технологий

Интернет является вершиной эволюции телекоммуникационных сетей, самой быстрорастущей технической системой в истории человечества. Интернет растет и качественно развивается постоянно, начиная с 80-х годов, и в соответствии с прогнозами специалистов этот процесс будет продолжаться.

«Размеры» интернета можно оценивать по-разному. Чаще всего используют такие показатели, как количество подключенных к нему терминальных устройств, количество пользователей и объем трафика, передаваемый в единицу времени.

Качественное развитие интернета проявляется в появлении все новых и новых сервисов, например, уже упомянутых интернет-вариантов телефонии и телевидения, а также новых типов терминальных устройств. Так, помимо «чистых» компьютеров к интернету стали подключаться разнообразные устройства со встроенными «компьютерами» — смартфоны, планшеты, бытовые приборы, автомобили, и этот список можно продолжать еще долго.

Рассмотрим в цифрах количественный рост интернета (www.statista.com).

На рис. 1.6 показан график роста числа пользователей интернета за 50 лет существования этой сети. В 2023 году их число по всему миру составило более 5,3 миллиарда, то есть 65 % населения земного шара.

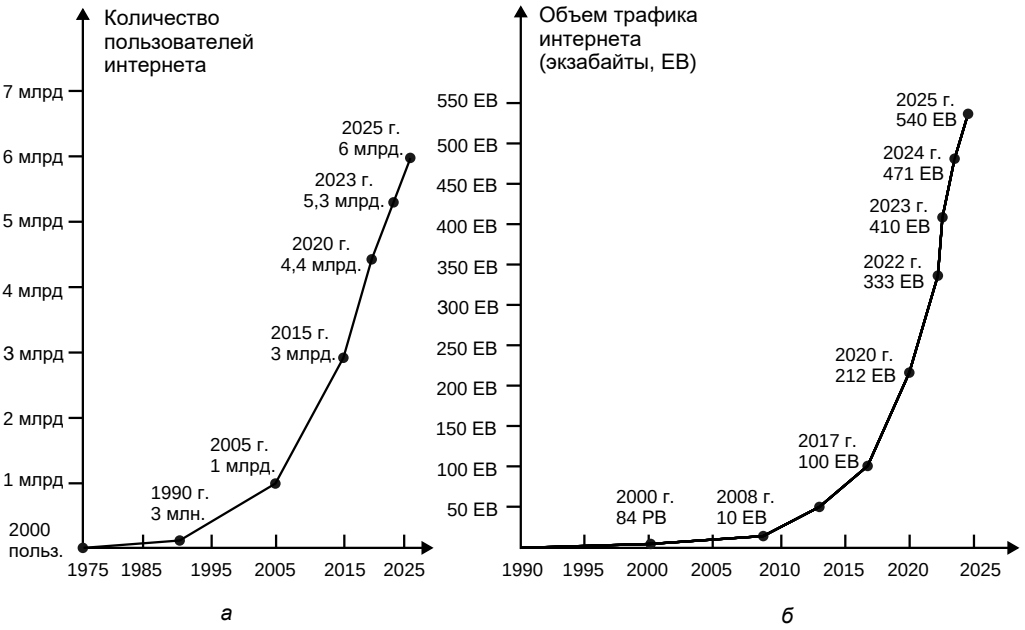


Рис. 1.6. Рост числа пользователей и трафика интернета

Количество терминальных устройств, выполняющих функции серверов (без учета пользовательских устройств) росло примерно такими же темпами: в 1980 году насчитывалось около 1000 хостов, подключенных к интернету, в 1991-м — более 1 000 000, в начале 2000-х — около 100 000 000 и, наконец, в 2018 году — свыше 1 миллиарда. С учетом пользовательских устройств (настольных компьютеров, ноутбуков, планшетов и мобильных телефонов) общее количество терминальных устройств, подключенных к интернету, составило в 2023 году около 27 миллиардов.

Рост трафика интернета в первых двух десятилетиях XXI века имел взрывной характер (на графике показано количество байтов, переданных в месяц через магистрали глобальной сети): с 84 РВ (петабайт) в 2000 году (1 петабайт = 1000 терабайт = 1 000 000 гигабайт) до 129 ЕВ (экзабайт) в 2018 году (1 экзабайт = 1000 петабайт), то есть более чем в 1000 раз.

Однако в последние годы рост трафика интернета хотя и продолжился, но скорость его роста замедлилась, и его уже нельзя назвать взрывным. Например, темпы роста фиксированного трафика доступа и мобильного трафика снизились с 20–30 % и более в год в начале 2010-х годов до примерно 10–15 % (фиксированная связь) и 5–10 % (мобильная связь) на зрелых рынках в 2024–2025 годах. Замедлился и рост числа пользователей интернета, а также некоторые другие показатели, что говорит о насыщении (или начале насыщения) рынка интернет-услуг.

Но интернет меняется не только количественно, но и качественно. Новые сервисы и новые типы терминальных устройств делали и делают интернет привлекательным для все большего числа массовых пользователей, которым раньше вряд ли пришло бы в голову использовать компьютер в повседневной жизни.

Если мы посмотрим на терминальные интернет-устройства, то увидим, что сегодня большую их часть составляют не традиционные персональные настольные компьютеры, а *мобильные устройства* — планшеты и смартфоны. Все большее распространение получает и такой тип терминальных устройств, как встроенные компьютеры. Они не так заметны, как планшеты и смартфоны, потому что работают внутри привычных систем и устройств, таких как отопительные системы, камеры слежения, телевизоры, холодильники или самоуправляемые автомобили. Такие компьютеры сами инициируют обмен данными между собой или со своими удаленными центрами управления через интернет, являясь, по существу, его пользователями. Этот новый класс пользователей породил и новый термин — **интернет вещей** (Internet of Things, IoT), подчеркивающий его отличие от традиционного интернета пользователей-людей.

В результате, если в 2013 году больше половины интернет-трафика (67 %) генерировали персональные компьютеры, то сейчас они уступили пальму первенства мобильным устройствам и встроенным компьютерам.

Существенно менялся и процентный состав приложений, генерирующих трафик. Так, если в 90-е годы и начале 2000-х в общем объеме преобладал трафик приложений, передающих файлы (файлы электронной почты, веб-страниц, музыки и кинофильмов), то уже к 2010 году он уступил лидерство трафику приложений, передающих видеопотоки в реальном масштабе времени (таких, как интернет-телевидение, показ кинофильмов в онлайновом режиме по требованию, видеоконференции).

Важной вехой на пути развития интернета стали **облачные вычисления**, которые позволяют разгрузить пользовательский компьютер и перенести хранение данных и выполнение приложений на некоторые удаленные компьютеры, связанные с пользовательским компьютером через сеть. Облачные вычисления стали еще одной причиной увеличения трафика интернета, так как пользователи этого сервиса постоянно обращаются к внешним серверам провайдера, вместо того чтобы производить вычисления и другие манипуляции с данными локально, на своих персональных компьютерах или на корпоративных серверах.

Появление новых терминальных устройств, новых приложений и услуг интернета привело к *повышению* требований к характеристикам сети.

Например, автомобили без водителя сейчас проходят испытания и скоро станут повседневностью. Интернет является неотъемлемой составной частью системы управления такими автомобилями, поддерживая обмен информацией в реальном времени между локальным компьютером автомобиля и удаленным приложением, решающим сложные задачи, возможно, с применением элементов искусственного интеллекта, которые не под силу локальному компьютеру с его ограниченными ресурсами. Понятно, что в этом случае время реакции сети и ее надежность являются критическими параметрами, иначе автомобиль просто станет опасным объектом на дороге. Новые требования к реакции сети могут повлечь за собой не только необходимость в разработке более скоростных сетевых технологий, но и изменения архитектуры интернета с перенесением центров вычислений ближе к его периферии.

Феноменальный рост и изменчивость интернета (в различных аспектах) оказывали и оказывают сильнейшее влияние на технологии компьютерных сетей, заставляя их постоянно изменяться и совершенствоваться, приспосабливаясь к новым требованиям пользователей и их количеству.

Искусственный интеллект в компьютерных сетях

В последние несколько лет наблюдается бум в области **искусственного интеллекта**, ИИ (Artificial Intelligence, AI). Термин ИИ/AI у всех на слуху, а производители в самых разных отраслях гордятся тем, что их новые продукты (даже винодельческие) «построены на основе AI».

Интерес к ИИ особенно усилился с предоставлением в ноябре 2022-го свободного доступа к системе ИИ ChatGPT, способной поддерживать общение на естественном языке и генерировать оригинальные тексты на различные темы, пользуясь обширной базой данных, созданной в процессе тренировки системы (около 300 миллиардов слов, 570 GB). Системы генерации изображений и музыки Runway, Pika Labs, Suno, AIVA и многие другие также вызывают огромный интерес.

В компьютерных сетях появились виртуальные сетевые помощники с элементами ИИ, которые применяются инженерами провайдеров интернета и администраторами корпоративных сетей для обнаружения и устранения проблем в сетях, например **Juniper Marvis AI Assistant** или **Cisco AI Assistant**. Системы ИИ используются при проектировании сетей, для прогноза роста и распределения трафика и оптимизации на основе этих данных структуры сети. В области безопасности системы ИИ распознают атаки на информационные и сетевые ресурсы и выдают рекомендации по их отражению. В общем, уже сегодня многие интеллектуальные работы по проектированию, развитию и обслуживанию сетей автоматизированы и отданы на откуп системам ИИ. Производители сетевого оборудования делают оптимистические прогнозы, обещая, что к 2030 году компьютерные сети станут полностью автоматическими, то есть для поддержания их работы человеческий труд не понадобится.

Ожидания от систем ИИ очень высокие. Кажется, что давние обитатели научно-фантастических книг — мыслящие компьютеры — вот-вот станут реальностью. Как-то один из инженеров Google объявил, что программа LaMDA, с которой он работал, обладает разумом, но был за это отстранен от работы.

Для трезвой оценки состояния и перспектив ИИ полезно посмотреть на эволюцию этой отрасли знаний. ИИ как отрасль науки и техники насчитывает многолетнюю историю, первые работы относятся к 30-м годам прошлого века. За это время системы ИИ пережили несколько периодов бурного расцвета и несколько периодов жестокой зимы. Одним из таких периодов активизации работ в области ИИ были 50-е годы. В 1951 году были разработаны первые программы, умеющие играть в шахматы и шашки, а в 1955 году была создана программа Logic Theorist, которая успешно доказала 38 из 52 математических теорем, приведенных в известной книге для математиков Рассела и Уайтхэда «Principia Matematica».

Появление такого рода «умных» программ породило дискуссию о том, может ли машина мыслить и что это такое — «искусственный интеллект». Дискуссия эта продолжается уже много лет и, к сожалению, не дает ясных результатов. Что такое «машина», «мышление», «разум», «сознание» — наука не дает четких определений этим базовым понятиям. Именно из-за этого один из основоположников теории вычислений, Алан Тьюринг, отказался от прямого определения искусственного интеллекта и предложил практический косвенный подход — **тест Тьюринга**: если жюри (публика, общество) не может отличить, кто из двух собеседников, с которыми жюри ведет разговор на естественном языке, является

компьютерной программой, а кто — человеком, то эта программа обладает интеллектом. Тест Тьюринга обладает как достоинствами (простота), так и недостатками (субъективность или некомпетентность жюри, узкая область сравнения способностей машины и человека). Из-за этих недостатков он редко применяется на практике. Но тест Тьюринга задал новый подход к оценке «умственных» возможностей компьютерных программ, подход, основанный на сравнении результатов работы человека и машины при решении конкретных задач.

Абсолютизируя этот критерий, можно утверждать, что любой результат работы компьютерной программы является проявлением интеллекта. Например, способность некоторого индивидуума умножать многозначные числа безусловно свидетельствует о наличии интеллекта у этого человека, и, следовательно, любой компьютер обладает интеллектом по этому критерию. Мы привели эту ремарку только для того, чтобы показать, насколько нечеткими являются любые определения ИИ и насколько важно проявлять здоровый скептицизм, оценивая интеллект компьютерной программы.

Другой крайностью является предъявление к машине очень высоких требований. Его сторонники считают, что машину можно считать мыслящей только в том случае, когда она создает оригинальные произведения, а не пусть и качественные, но компиляции созданных ранее человеческих творений. Эта точка зрения была высказана более 150 лет назад первой в мире программисткой Адой Лавлейс после ее опыта по программированию Аналитической машины Бэббиджа: «Аналитическая машина не претендует на создание чего-либо оригинального. Она может делать только то, что мы говорим ей делать. Ее роль состоит в том, чтобы помогать нам делать то, с чем мы уже знакомы».

Теоретическая дискуссия о потенциале систем ИИ шла параллельно с их развитием, и в конце 60-х годов наступило разочарование, которое часто происходит после периода завышенных ожиданий. Системы ИИ 60-х годов не достигли каких-либо впечатляющих практических результатов, они скорее представляли собой «игрушки», а не коммерческие продукты (например, программа-переводчик со словарем в несколько сотен слов), возможно, из-за очень ограниченных возможностей тогдашних компьютеров. Так или иначе, а разочарование было столь сильным, что на 15 лет область ИИ была вытеснена из списка перспективных информационных технологий.

В 80-е годы наступил новый период расцвета. Воплощением первого коммерческого успеха идей ИИ стали **экспертные системы**, которые работали на основе опыта, накопленного экспертами в определенной узкой области знаний, и принимали решения в соответствии с набором небольшого числа правил. Например, система MYCIN успешно диагностировала бактериальные инфекции на основе результатов анализа крови и других медицинских данных и выписывала нужный антибиотик. Но их успех был кратковременным. Такие системы плохо масштабировались и модифицировались, они не умели самообучаться, и после краха этой отрасли ИИ снова пережил длительный период забвения.

Сегодняшние системы ИИ являются результатом постепенного — и не афишируемого — прогресса в этой области, происходившего «за сценой» последние 10–15 лет. Этот прогресс можно было заметить во многих программных продуктах и системах. Например, поисковые системы, такие как Google и Microsoft Bing, стали за эти годы гораздо «интеллектуальнее», давая по контексту запроса тот ответ, который и был нужен пользователю, а не просто кусочки текста, в которых встречается заданное ключевое слово. Радикально повысилось и качество перевода с одного языка на другой. Виртуальные помощники, такие как Apple Siri, Amazon Alexa, Яндекс Алиса и др. стали поддерживать общение

на естественном для человека языке на довольно хорошем уровне. Но поскольку в этот период при описании новых свойств продуктов термин ИИ/AI, как правило, не использовался, то пользователи теперь могут воскликнуть подобно герою Мольера господину Журдену: «Я и не подозревал, что вот уже более ~~сорока лет говорю прозой~~! Десять лет работаю с системами ИИ». Примеров «умных» программ и систем 2000–2010-х годов, которые до поры до времени не выпячивали свою принадлежность к миру ИИ, можно привести множество.

Большую роль в прогрессе систем ИИ в эти годы, особенно в системах, ведущих диалог с пользователем на естественном для человека языке, сыграли нейронные сети, или нейросети. Искусственные нейросети как идея были предложены еще в 1943 году, а первые реализации нейронных сетей появились в 50-е годы. Искусственные нейросети имитируют в очень упрощенной форме работу мозга человека, состоящего из более чем 80 миллиардов нейронов, объединенных в различные области и соединенных друг с другом связями самой разнообразной топологии — иерархическими, рекурсивными, внутриобластными, межобластными и т. д. Нейроны мозга принимают сигналы (информацию) от других нейронов, и в том случае, если суммарный входной сигнал превышает некоторый порог, передают ее дальше, что в конечном итоге приводит к реакции мозга — например, восприятию зрительного образа или принятию некоторого решения.

Искусственные нейроны имитируют такие процессы, принимая в качестве входных сигналов числа и преобразуя их в выходное значение, используя при этом некоторую пороговую функцию (аналог порогового поведения нейронов мозга). Нейроны самого нижнего слоя сети принимают на свои входы исходные данные той задачи, которую сеть должна решить, — например, в случае задачи распознавания образов, пикселы изображения. Выходное значение самого верхнего слоя сети представляет собой решение этой задачи, например, сеть считает, что изображение с вероятностью 90 % является изображением кошки. Значимость вклада выходного сигнала каждого нейрона в конечный результат зависит от весового коэффициента, в соответствии со значением которого этот выходной сигнал передается следующему нейрону.

Для того чтобы сеть смогла решать задачи, ее нужно **тренировать** (обучать). Обучение происходит на большом количестве наборов исходных данных некоторой задачи, для которых известен желательный результат работы сети. Например, эти наборы могут состоять из большого количества изображений различных животных, среди которых есть кошки. Желательный результат — распознавание изображений, на которых есть кошка, независимо от того, в каком ракурсе и масштабе она там присутствует (излюбленный прием создателей CAPTCHA, которые считают, что простые боты не могут справиться с такой задачей). Обучение состоит в подборе такого сочетания весовых коэффициентов связей между нейронами сети, которое дает максимальное приближение к желаемому результату, то есть к многократному решению задачи оптимизации большой размерности. Обучение нейросети сложным задачам обычно требует очень больших усилий — ручной подготовки «помеченных» (то есть сопровождающихся указанием желательного результата) наборов исходных данных и прогону их через нейросеть. Многие современные системы ИИ сочетают обучение под руководством человека с самообучением. Широкие возможности для самообучения сулит доступ систем ИИ ко всей информационной базе интернета, и такой доступ имеется у некоторых развитых систем уже сегодня.

Первые искусственные нейронные сети были достаточно простыми, состояли из нескольких сотен нейронов, объединенных в три-четыре слоя. Они могли решать ограниченный

круг задач, например распознавать несколько устойчивых сочетаний исходных данных. В течение долгого времени их потенциал по разным причинам не был использован, и только сравнительно недавно они заняли центральное место в системах ИИ.

Несмотря на то что системы ИИ нового поколения базируются на принципах, моделях и алгоритмах, известных достаточно давно, сейчас мы наблюдаем *переход количества в качество*, и это новое качество обещает быть поистине революционным. Действительно, количественные характеристики сегодняшних систем ИИ — это миллиарды (и даже триллионы) искусственных нейронов, сотни слоев иерархии нейронной сети, сотни тысяч обработанных документов и произведений человеческого гения и тысячи и тысячи человеко-часов, затраченных при тренировке систем ИИ. Понятно, что существование таких «монстров» было невозможно ни в 60-е, ни даже в 90-е годы — вычислительная мощность компьютеров тех лет была абсолютно недостаточна для их работы. Появление облачных центров с тысячами и даже десятками тысяч серверов изменило ситуацию, создав основу для успешной работы сверхбольших моделей ИИ в реальном времени.

ГЛАВА 2 Общие принципы построения сетей

Простейшая сеть из двух компьютеров

Совместное использование ресурсов

Исторически главной целью объединения компьютеров в сеть было *разделение ресурсов*: пользователи компьютеров, подключенных к сети, или приложения, выполняемые на этих компьютерах, получают возможность автоматического доступа к разнообразным ресурсам остальных компьютеров сети, к числу которых относятся:

- периферийные устройства, такие как диски, принтеры, плоттеры, сканеры и др.;
- данные, хранящиеся в оперативной памяти или на внешних запоминающих устройствах;
- вычислительная мощность (за счет удаленного запуска «своих» программ на «чужих» компьютерах).

Чтобы обеспечить пользователей разных компьютеров возможностью совместного использования ресурсов сети, компьютеры необходимо оснастить некими дополнительными *сетевыми* средствами.

Рассмотрим простейшую сеть, состоящую из двух компьютеров, к одному из которых подключен принтер (рис. 2.1). Какие дополнительные средства должны быть предусмотрены в обоих компьютерах, чтобы с принтером мог работать не только пользователь компьютера *B*, к которому этот принтер непосредственно подключен, но и пользователь компьютера *A*?

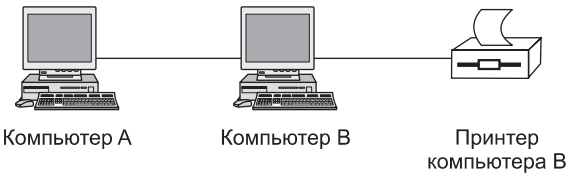


Рис. 2.1. Простейшая сеть

Сетевые интерфейсы

Для связи устройств в них прежде всего должны быть предусмотрены внешние интерфейсы.

Интерфейс — в широком смысле — формально определенная логическая и/или физическая граница между отдельными объектами, которые обмениваются информацией. Интерфейс задает параметры, процедуры и характеристики взаимодействия объектов.

ПРИМЕЧАНИЕ

Наряду с *внешними* электронные устройства могут использовать *внутренние* интерфейсы, определяющие логические и физические границы между входящими в их состав модулями: оперативной памятью, процессором и др.

Разделяют физический и логический интерфейсы.

- **Физический интерфейс** (называемый также **портом**) определяется набором электрических связей и характеристиками сигналов. Обычно он представляет собой разъем с набором контактов, каждый из которых имеет определенное назначение, например, это может быть группа контактов для передачи данных, контакт синхронизации данных и т. п. Пара разъемов соединяется **кабелем**, состоящим из набора проводов, каждый из которых соединяет соответствующие контакты. В таких случаях говорят о создании **линии**, или **канала**, **связи** между двумя устройствами.
- **Логический интерфейс** (называемый также **протоколом**) — это набор информационных сообщений определенного формата, которыми обмениваются два устройства или две программы, а также набор правил, определяющих логику обмена этими сообщениями.

На рис. 2.2 мы видим интерфейсы двух типов: «компьютер — компьютер» и «компьютер — периферийное устройство».

- *Интерфейс «компьютер — компьютер»* позволяет двум компьютерам обмениваться информацией. С каждой стороны он реализуется парой:
  - аппаратным модулем, называемым **сетевым адаптером** или сетевой **интерфейсной картой** (ИК), или (в англоязычном варианте) Network Interface Card, NIC;
  - **драйвером сетевой интерфейсной карты** — специальной программой, управляющей работой сетевой интерфейсной карты.

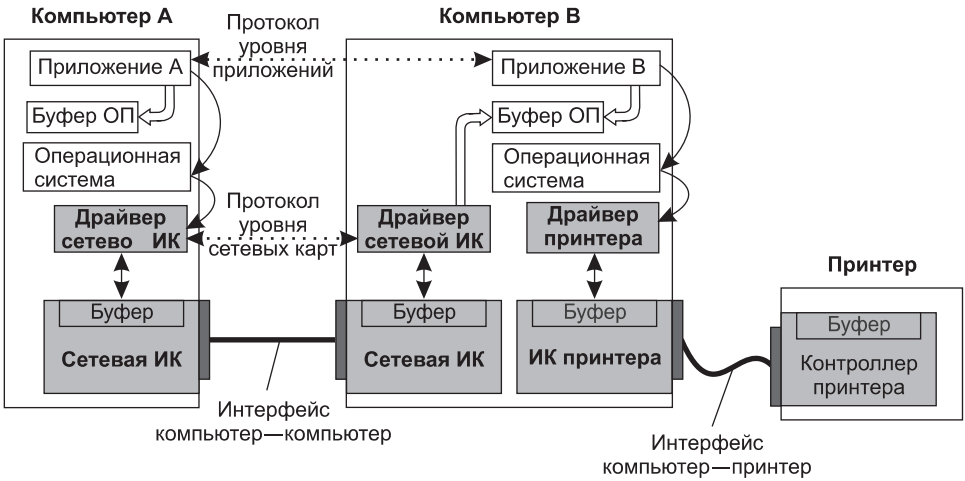


Рис. 2.2. Совместное использование принтера в компьютерной сети

- *Интерфейс «компьютер — периферийное устройство»* (в данном случае интерфейс «компьютер — принтер») позволяет компьютеру управлять работой периферийного устройства (ПУ). Этот интерфейс реализуется:
  - со стороны компьютера — интерфейсной картой и драйвером ПУ (принтера), подобным сетевой интерфейсной карте и ее драйверу;
  - со стороны ПУ — контроллером ПУ (принтера), обычно представляющим собой аппаратное устройство[1], принимающее от компьютера как *данные*, например, байты информации, которую нужно распечатать на бумаге, так и *команды*, которые он отрабатывает, управляя электромеханическими частями периферийного устройства, например, выталкивая лист бумаги из принтера или перемещая магнитную головку диска.

Связь компьютера с периферийным устройством

Для понимания того, как решить задачу организации доступа приложения, выполняемого на компьютере *A*, к ПУ через сеть, давайте прежде всего посмотрим, как управляет этим устройством приложение, выполняемое на компьютере *B*, к которому данное ПУ подключено непосредственно (см. рис. 2.2).

1. Пусть приложению *B* в какой-то момент потребовалось вывести на печать некоторые данные. Для этого приложение обращается с запросом на выполнение операции ввода-вывода к *операционной системе* (как правило, драйвер не может быть запущен на выполнение непосредственно приложением). В запросе указываются адрес данных, которые необходимо напечатать (адрес буфера ОП), и информация о том, на каком периферийном устройстве эту операцию требуется выполнить.
2. Получив запрос, операционная система запускает программу — *драйвер принтера*. С этого момента все дальнейшие действия по выполнению операции ввода-вывода со стороны компьютера реализуются только драйвером принтера и работающим под его управлением аппаратным модулем — *интерфейсной картой принтера* без участия приложения и операционной системы.
3. Драйвер принтера оперирует командами, понятными контроллеру принтера, такими, например, как «Печать символа», «Перевод строки», «Возврат каретки». Драйвер в определенной последовательности загружает коды этих команд, а также данные, взятые из буфера ОП, в буфер интерфейсной карты принтера, которая побайтно передает их по сети контроллеру принтера.
4. Интерфейсная карта выполняет низкоуровневую работу, не вдаваясь в детали, касающиеся логики управления устройством, смысла данных и команд, передаваемых ей драйвером, считая их однородным потоком байтов. После получения от драйвера очередного байта интерфейсная карта просто последовательно передает биты в линию связи, представляя каждый бит электрическим сигналом. Чтобы контроллеру принтера стало понятно, что начинается передача байта, перед передачей первого бита информационная карта формирует **стартовый сигнал** специфической формы, а после передачи последнего информационного бита — **стоповый сигнал**. Эти сигналы синхро-

[1] Встречаются и программно-управляемые контроллеры, например для управления современными принтерами, обладающими сложной логикой.

низируют передачу байта. Контроллер, опознав стартовый бит, начинает принимать информационные биты, формируя из них байт в своем приемном буфере. Помимо информационных битов карта может передавать бит контроля четности для повышения достоверности обмена. При корректно выполненной передаче в буфере принтера устанавливается соответствующий признак.

5. Получив очередной байт, контроллер интерпретирует его и запускает заданную операцию принтера. Закончив работу по печати всех символов документа, драйвер принтера сообщает операционной системе о выполнении запроса, а та, в свою очередь, сигнализирует об этом событии приложению.

Обмен данными между двумя компьютерами

Механизмы взаимодействия компьютеров в сети многое позаимствовали у схемы взаимодействия компьютера с периферийными устройствами. В самом простом случае связь компьютеров может быть реализована с помощью тех же самых средств, которые используются для связи компьютера с периферией, с той разницей, что в этом случае активную роль играют обе взаимодействующие стороны.

Приложения *A* и *B* (см. рис. 2.2) управляют процессом передачи данных путем обмена **сообщениями**. Чтобы приложения могли «понимать» получаемую друг от друга информацию, программисты, разрабатывавшие эти приложения, должны *строго оговорить* форматы и последовательность сообщений, которыми приложения будут обмениваться во время выполнения этой операции. Например, они могут договориться о том, что любая операция обмена данными начинается с передачи сообщения, запрашивающего информацию о готовности приложения *B*; что в следующем сообщении идут идентификаторы компьютера и пользователя, сделавшего запрос; что признаком срочного завершения операции обмена данными является определенная кодовая комбинация, и т. п. Тем самым определяется **протокол взаимодействия приложений** для выполнения операции данного типа.

Аналогично тому, как при выводе данных на печать необходимо передавать принтеру дополнительно некоторый объем служебной информации — в виде команд управления принтером, так и здесь для передачи данных из одного компьютера в другой необходимо сопровождать эти данные дополнительной информацией в виде протокольных сообщений, которыми обмениваются приложения.

Заметим, что для реализации протокола нужно, чтобы к моменту возникновения потребности в обмене данными были активны оба приложения: как приложение *A*, которое посылает инициирующее сообщение, так и приложение *B*, которое должно быть готово принять это сообщение и выработать реакцию на него.

Передача любых данных (как сообщений протокола приложений, так и собственно данных, составляющих цель операции обмена) происходит в соответствии с одной и той же процедурой.

На стороне компьютера A приложение, следуя логике протокола, размещает в буфере ОП либо собственное очередное сообщение, либо данные и обращается к ОС с запросом на выполнение операции межкомпьютерного обмена данными. ОС запускает соответствующий драйвер сетевой карты, который загружает байт из буфера ОП в буфер интерфейсной карты, после чего инициирует ее работу. Сетевая интерфейсная карта последовательно передает биты в линию связи, дополняя каждый новый байт стартовым и стоповым битами.

На стороне компьютера B сетевая интерфейсная карта принимает биты, поступающие со стороны внешнего интерфейса, и помещает их в собственный буфер. После того как получен стоповый бит, интерфейсная карта устанавливает признак завершения приема байта и выполняет проверку корректности приема, например, путем контроля бита четности. Факт корректного приема байта фиксируется драйвером сетевой интерфейсной карты компьютера *B*. Драйвер переписывает принятый байт из буфера интерфейсной карты в заранее зарезервированный буфер ОП компьютера *B*. Приложение *B* извлекает данные из буфера и интерпретирует их в соответствии со своим протоколом либо как сообщение, либо как данные. Если согласно протоколу приложение *B* должно передать ответ приложению *A*, то выполняется симметричная процедура.

Таким образом, связав электрически и информационно два автономно работающих компьютера, мы получили простейшую **компьютерную сеть**.

Доступ к периферийным устройствам через сеть

Итак, мы имеем в своем распоряжении механизм, который позволяет приложениям, выполняющимся на разных компьютерах, обмениваться данными. И хотя приложение *A* (см. рис. 2.2) по-прежнему не может управлять принтером, подключенным к компьютеру *B*, оно может теперь воспользоваться средствами межкомпьютерного обмена данными, чтобы передать приложению *B* «просьбу» выполнить для него требуемую операцию. Приложение *A* должно «объяснить» приложению *B*, какую операцию необходимо выполнить, с какими данными, на каком из имеющихся в его распоряжении устройств, в каком виде должен быть распечатан текст, и т. п. В ходе печати могут возникнуть ситуации, о которых приложение *B* должно оповестить приложение *A*, например об отсутствии бумаги в принтере. То есть для решения поставленной задачи — доступа к принтеру по сети — должен быть разработан специальный протокол взаимодействия приложений *A* и *B*.

А теперь посмотрим, как работают вместе все элементы этой простейшей компьютерной сети при решении задачи совместного использования принтера.

1. В соответствии с принятым протоколом приложение *A* формирует сообщение-запрос к приложению *B*, помещает его в буфер ОП компьютера *A* и обращается к ОС, снабжая ее необходимой информацией.
2. ОС запускает драйвер сетевой интерфейсной карты, сообщая ему адрес буфера ОП, где хранится сообщение.
3. Драйвер и сетевая интерфейсная карта компьютера *A*, взаимодействуя с драйвером и интерфейсной картой компьютера *B*, передают сообщение байт за байтом в буфер ОП компьютера *B*.
4. Приложение *B* извлекает сообщение из буфера, интерпретирует его в соответствии с протоколом и выполняет необходимые действия. В число таких действий входит, в том числе, обращение к ОС с запросом на выполнение тех или иных операций с локальным принтером.
5. ОС запускает драйвер принтера, который в кооперации с интерфейсной картой и контроллером принтера выполняет требуемую операцию псчати.

Уже на этом начальном этапе, рассматривая связь компьютера с периферийным устройством, мы столкнулись с важнейшими «сетевыми» понятиями: интерфейсом и протоколом,

драйвером и интерфейсной картой, а также с проблемами, характерными для компьютерных сетей: согласованием интерфейсов, синхронизацией асинхронных процессов, обеспечением достоверности передачи данных.

Сетевое программное обеспечение

Мы только что рассмотрели случай совместного использования принтера в простейшей сети, состоящей только из двух компьютеров. Однако даже на этом начальном этапе мы уже можем сделать некоторые выводы относительно строения сетевого программного обеспечения: сетевых служб, сетевой операционной системы и сетевых приложений.

Сетевые службы и сервисы

Потребность в доступе к удаленному принтеру может возникать у пользователей самых разных приложений: текстового редактора, графического редактора, системы управления базой данных (СУБД). Очевидно, что дублирование в каждом из приложений общих для всех них функций по организации удаленной печати является избыточным.

Более эффективным представляется подход, при котором эти функции исключаются из приложений и оформляются в виде пары специализированных программных модулей — *клиента* и *сервера печати* (рис. 2.3), функции которых ранее выполнялись, соответственно, приложениями *A* и *B*. Теперь эта пара «клиент — сервер» может быть использована любым приложением, выполняемым на компьютере *A*.

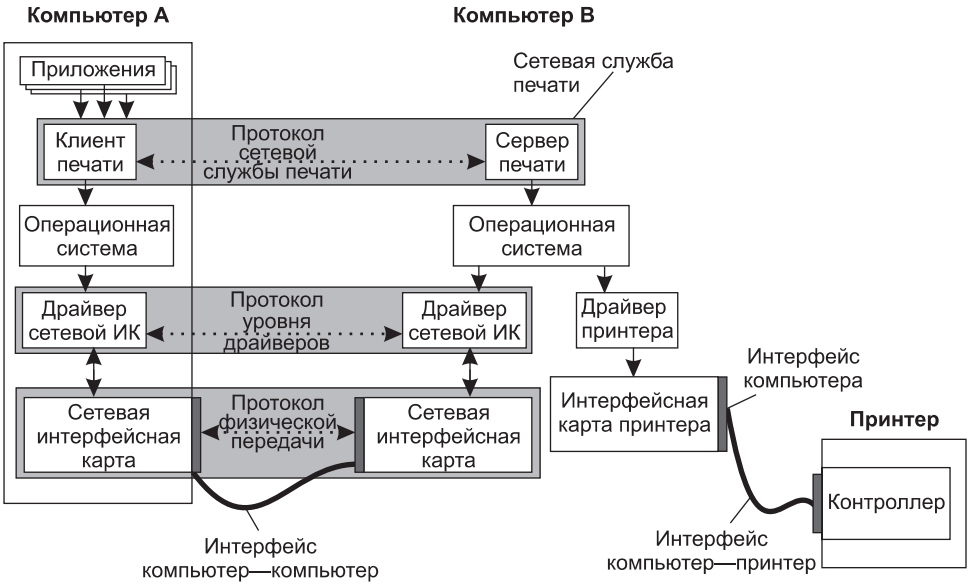


Рис. 2.3. Совместное использование принтера в компьютерной сети с помощью сетевой службы печати

Обобщая такой подход применительно к другим типам разделяемых ресурсов, дадим следующие определения[1]:

Клиент — это модуль, предназначенный для формирования и передачи сообщений-запросов к ресурсам удаленного компьютера от разных приложений с последующим приемом результатов из сети и передачей их соответствующим приложениям.

Сервер — это модуль, который постоянно ожидает прихода из сети запросов от клиентов и, приняв запрос, пытается его обслужить, как правило, с участием локальной ОС; один сервер может обслуживать запросы сразу нескольких клиентов (поочередно или одновременно).

Пара клиент-сервер, предоставляющая доступ к конкретному типу ресурса компьютера через сеть, образует **сетевую службу**.

Каждая служба связана с определенным типом сетевых ресурсов. Так, на рис. 2.3 модули клиента и сервера, реализующие удаленный доступ к принтеру, образуют сетевую службу печати.

Среди сетевых служб можно выделить такие, которые ориентированы не на простого пользователя, как, например, файловая служба или служба печати, а на администратора. Такие службы направлены на организацию работы сети. Например, **справочная служба** или **служба каталогов** предназначена для ведения базы данных о пользователях сети, обо всех ее программных и аппаратных компонентах.

Услуги, предоставляемые службой, называются **сервисом**.

Служба может предоставлять сервис одного или нескольких типов. Так, к числу услуг, оказываемых справочной службой, помимо учета ресурсов, относятся сервисы аудита, аутентификации, авторизации и др.

Для поиска и просмотра информации в интернете используется **веб-служба**, состоящая из **веб-сервера** и клиентской программы, называемой **веб-браузером** (web browser). Разделяемым ресурсом в данном случае является **веб-сайт** — определенным образом организованный набор файлов, содержащих связанную в смысловом отношении информацию и хранящихся на внешнем накопителе веб-сервера.

На схеме веб-службы, показанной на рис. 2.4, два компьютера связаны не непосредственно, как это было во всех предыдущих примерах, а через множество промежуточных компьютеров и других сетевых устройств, входящих в состав интернета. Чтобы отразить этот факт графически, мы поместили между двумя компьютерами так называемое **коммуникационное облако**, позволяющее абстрагироваться от всех деталей среды передачи сообщений. Обмен сообщениями между клиентской и серверной частями веб-службы выполняется по стандартному протоколу HTTP и никак не зависит от того, передаются ли эти сообщения «из рук в руки» (от интерфейса одного компьютера к интерфейсу другого) или через большое число посредников — транзитных коммуникационных устройств.

[1] Сервером и клиентом также называют компьютеры, на которых работают серверная и клиентская части сетевой службы соответственно.

Вместе с тем усложнение среды передачи сообщений приводит к возникновению новых дополнительных задач, на решение которых не был рассчитан упоминавшийся ранее простейший драйвер сетевой интерфейсной карты. Вместо него на взаимодействующих компьютерах должны быть установлены более развитые программные **транспортные средства**.

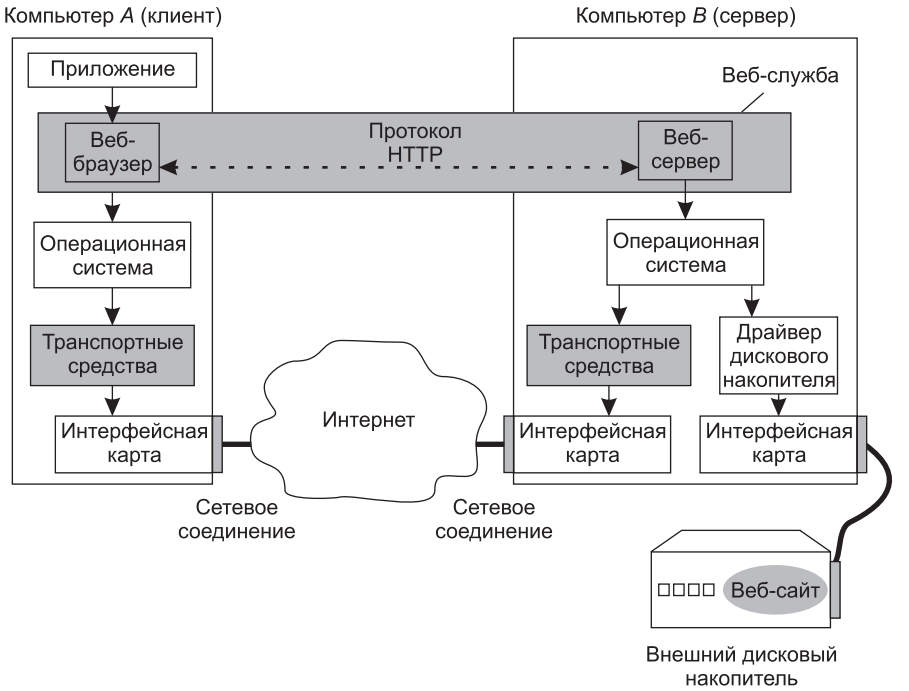


Рис. 2.4. Веб-служба

Сетевая операционная система

Операционную систему компьютера часто определяют как взаимосвязанный набор системных программ, который обеспечивает эффективное управление ресурсами компьютера (памятью, процессором, внешними устройствами, файлами и др.), а также предоставляет пользователю удобный интерфейс для работы с аппаратурой компьютера и разработки приложений.

Говоря о *сетевой* ОС, мы, очевидно, должны расширить границы управляемых ресурсов за пределы одного компьютера.

Сетевой операционной системой называют операционную систему компьютера, которая, помимо управления локальными ресурсами, предоставляет пользователям и приложениям возможность эффективного и удобного доступа к информационным и аппаратным ресурсам других компьютеров сети.

Сегодня практически все операционные системы являются сетевыми.

Из примеров, рассмотренных в предыдущих разделах (см. рис. 2.3 и 2.4), мы видим, что удаленный доступ к сетевым ресурсам обеспечивается:

- сетевыми службами;
- средствами транспортировки сообщений по сети (в простейшем случае — сетевыми интерфейсными картами и их драйверами).

Следовательно, именно эти функциональные модули должны быть добавлены к ОС, чтобы она могла называться сетевой (рис. 2.5).

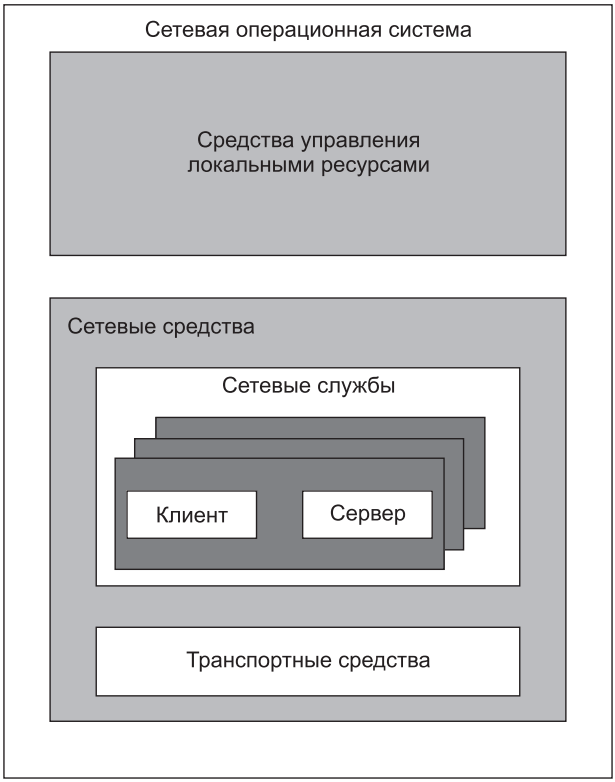


Рис. 2.5. Функциональные компоненты сетевой ОС

От того, насколько богатый набор сетевых служб и услуг предлагает операционная система конечным пользователям, приложениям и администраторам сети, зависит ее позиция в общем ряду сетевых ОС.

Помимо сетевых служб сетевая ОС включает *программные коммуникационные* (*транспортные*) *средства*, обеспечивающие совместно с аппаратными коммуникационными средствами передачу сообщений, которыми обмениваются клиентские и серверные части сетевых служб. Задачу коммуникации между компьютерами сети решают драйверы и протокольные модули. Они выполняют такие функции, как формирование сообщений, разбиение сообщения на части (пакеты, кадры), преобразование имен компьютеров в числовые

адреса, дублирование сообщений в случае их потери, определение маршрута в сложной сети и т. д.

И сетевые службы, и транспортные средства могут являться неотъемлемыми (встроенными) компонентами ОС или существовать в виде отдельных программных продуктов. Например, сетевая файловая служба обычно встраивается в ОС, а вот веб-браузер чаще всего является отдельным приложением. Типичная сетевая ОС имеет в своем составе широкий набор драйверов и протокольных модулей, однако у пользователя, как правило, есть возможность дополнить этот стандартный набор необходимыми ему программами. Решение о способе реализации клиентов и серверов сетевой службы, а также драйверов и протокольных модулей принимается разработчиками с учетом самых разных соображений: технических, коммерческих и даже юридических. Так, например, именно на основании антимонопольного закона США компании Microsoft было запрещено включать ее браузер Internet Explorer в состав ОС этой компании.

Сетевая служба может быть представлена в ОС либо обеими (клиентской и серверной) частями, либо только одной из них.

В первом случае операционная система, называемая **одноранговой**, не только позволяет обращаться к ресурсам других компьютеров, но и предоставляет собственные ресурсы в распоряжение пользователей других компьютеров. Например, если на всех компьютерах сети установлены и клиенты, и серверы файловой службы, то все пользователи сети могут совместно использовать файлы друг друга. Компьютеры, совмещающие функции клиента и сервера, называют одноранговыми узлами.

Операционная система, которая содержит преимущественно клиентские части сетевых служб, называется **клиентской**. Клиентские ОС устанавливаются на компьютеры, обращающиеся с запросами к ресурсам других компьютеров сети. За такими компьютерами, также называемыми клиентскими, работают рядовые пользователи. Обычно клиентские компьютеры относятся к классу относительно простых устройств.

К другому типу операционных систем относится **серверная ОС** — она ориентирована на обработку запросов из сети к ресурсам своего компьютера и включает в себя в основном серверные части сетевых служб. Компьютер с установленной на нем серверной ОС, занимающийся исключительно обслуживанием запросов других компьютеров, называют **выделенным сервером** сети. За выделенным сервером, как правило, обычные пользователи не работают.

ПРИМЕЧАНИЕ

Подробнее о сетевых операционных системах и встроенных в них сетевых службах вы можете прочитать в специальной литературе, а также в учебнике авторов «Сетевые операционные системы». Наиболее популярные сетевые службы Интернета, такие как электронная почта, веб-служба, IP-телефония и др., рассматриваются далее в части VII этой книги.

Сетевые приложения

На компьютере, подключенном к сети, могут запускаться приложения нескольких типов:

- **Локальное приложение** целиком выполняется на данном компьютере и использует только локальные ресурсы (рис. 2.6, *а*). Для такого приложения не требуется никаких сетевых средств, и оно может быть выполнено на автономно работающем компьютере.

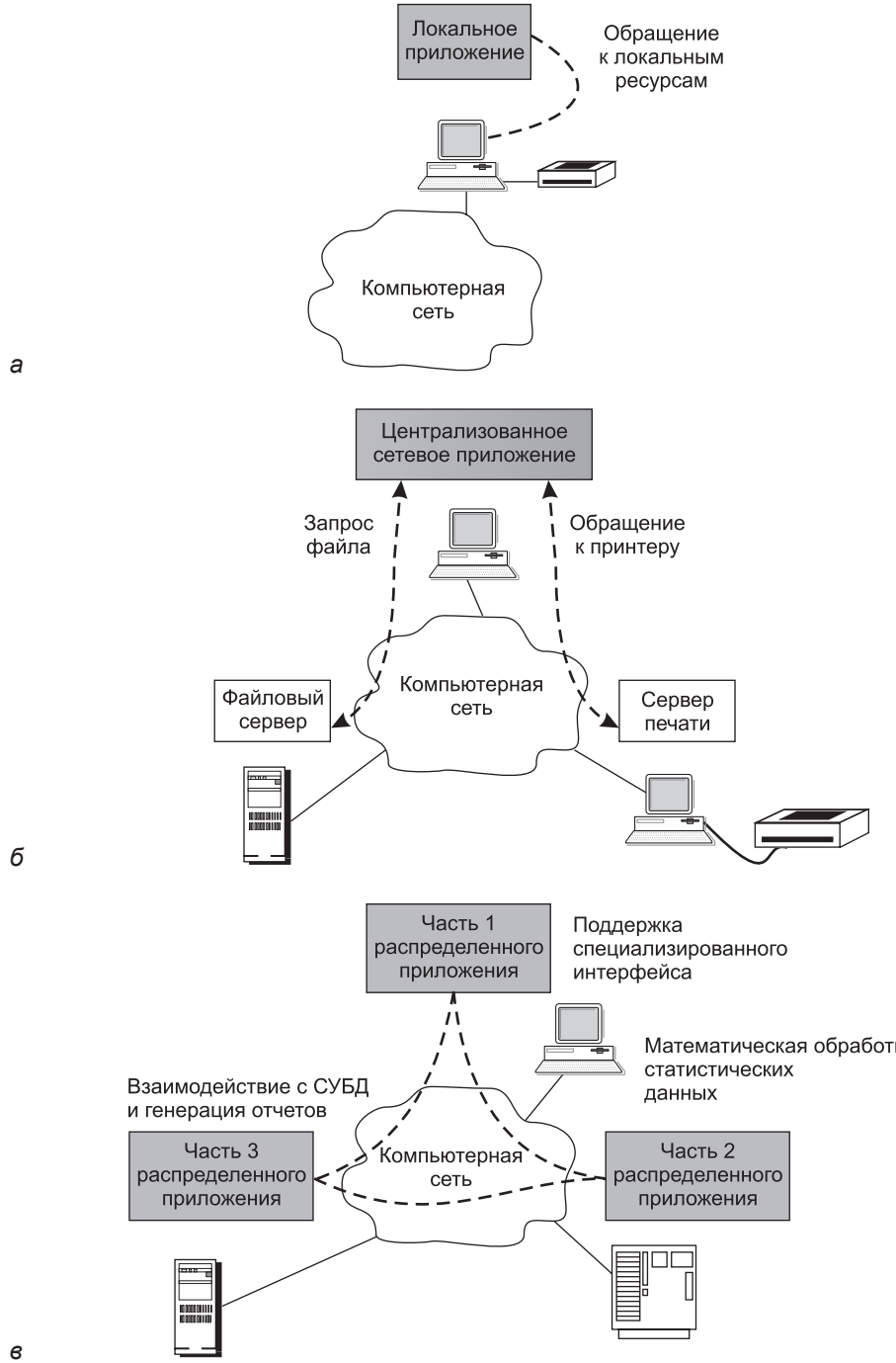


Рис. 2.6. Типы приложений, выполняющихся в сети

- **Централизованное сетевое приложение** целиком выполняется на данном компьютере, но обращается в процессе своей работы к ресурсам других компьютеров сети. В примере на рис. 2.6, *б* приложение, которое выполняется на клиентском компьютере, обрабатывает данные из файла, хранящегося на файл-сервере, а затем распечатывает результаты на принтере, подключенном к серверу печати. Очевидно, что работа такого типа приложений невозможна без участия сетевых служб и средств транспортировки сообщений.
- **Распределенное (сетевое) приложение** состоит из нескольких взаимодействующих частей, каждая из которых выполняет какую-то определенную законченную работу по решению прикладной задачи, причем каждая часть может выполняться и, как правило, выполняется на отдельном компьютере сети (рис. 2.6, *в*). Части распределенного приложения взаимодействуют друг с другом, используя сетевые службы и транспортные средства ОС. Распределенное приложение в общем случае имеет доступ ко всем ресурсам компьютерной сети.

Очевидным преимуществом распределенных приложений является возможность распараллеливания вычислений, а также специализация компьютеров. Так, в приложении, предназначенном, скажем, для анализа климатических изменений, можно выделить три достаточно самостоятельные части (см. рис. 2.6, *в*), допускающие распараллеливание. Первая часть приложения, выполняющаяся на сравнительно маломощном персональном компьютере, могла бы поддерживать специализированный графический пользовательский интерфейс, вторая — заниматься статистической обработкой данных на высокопроизводительном мейнфрейме, третья — генерировать отчеты на сервере с установленной стандартной СУБД. В общем случае каждая из частей распределенного приложения может быть представлена несколькими копиями, работающими на разных компьютерах. Скажем, в данном примере первую часть, ответственную за поддержку специализированного пользовательского интерфейса, можно было бы запустить на нескольких персональных компьютерах, что позволило бы работать с этим приложением нескольким пользователям одновременно.

Однако, чтобы добиться всех тех преимуществ, которые сулят распределенные приложения, их разработчикам приходится решать множество проблем, например: на сколько частей следует разбить приложение, какие функции возложить на каждую часть, как организовать взаимодействие этих частей, чтобы в случае сбоев и отказов оставшиеся части корректно завершали работу, и т. д. и т. п.

Заметим, что все сетевые службы, включая файловую службу, службу печати, службу электронной почты, службу удаленного доступа, интернет-телефонию и др., по определению относятся к классу распределенных приложений. Действительно, любая сетевая служба включает в себя клиентскую и серверную части, которые могут выполняться и обычно выполняются на разных компьютерах.

На рис. 2.7, иллюстрирующем распределенный характер веб-службы, мы видим различные виды клиентских устройств — персональные компьютеры, ноутбуки и мобильные телефоны — с установленными на них веб-браузерами, которые взаимодействуют по сети с веб-сервером. Таким образом, с одним и тем же веб-сайтом может одновременно работать множество — сотни и тысячи — сетевых пользователей.

Многочисленные примеры распределенных приложений можно встретить и в такой области, как обработка данных научных экспериментов. Это не удивительно, так как многие эксперименты порождают такие большие объемы данных, генерируемых в реальном

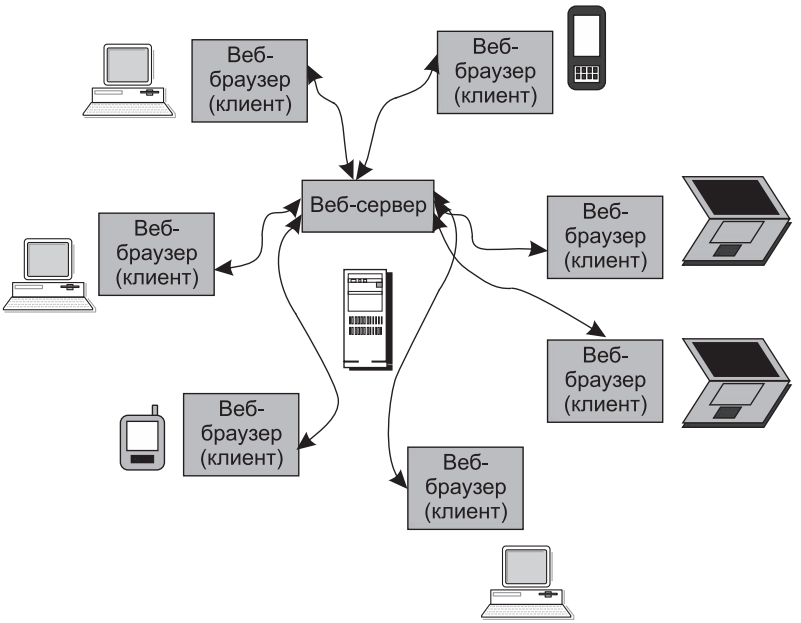


Рис. 2.7. Веб-служба как распределенное приложение

масштабе времени, которые просто невозможно обработать на одном, даже очень мощном суперкомпьютере. Кроме того, алгоритмы обработки экспериментальных данных часто легко распараллеливаются, что также важно для успешного применения взаимосвязанных компьютеров с целью решения какой-либо общей задачи. Одним из известных примеров распределенного научного приложения является программное обеспечение обработки данных большого адронного коллайдера (Large Hadron Collider, LHC), запущенного 10 сентября 2008 года в CERN, — это приложение работает более чем на 30 тысячах компьютеров, объединенных в сеть.

Физическая передача данных по линиям связи

Даже при рассмотрении простейшей сети, состоящей всего из двух машин, можно выявить многие проблемы, связанные с физической передачей сигналов по линиям связи.

Кодирование

В вычислительной технике для представления данных используется **двоичный код**. Внутри компьютера единицам и нулям данных соответствуют дискретные электрические сигналы.

Представление данных в виде электрических или оптических сигналов называется **кодированием**.

Существуют различные способы кодирования двоичных цифр, например **потенциальный способ**, при котором единице соответствует один уровень напряжения, а нулю — другой, или **импульсный способ**, когда для представления цифр используются импульсы различной полярности.

Аналогичные подходы применимы для кодирования данных и при передаче их между двумя компьютерами **по линиям связи**. Однако эти линии связи отличаются по своим характеристикам от линий внутри компьютера. Главное отличие внешних линий связи от внутренних состоит в их гораздо большей протяженности, а также в том, что они проходят вне экранированного корпуса по пространствам, зачастую подверженным воздействию сильных электромагнитных помех. Все это приводит к существенно бóльшим искажениям прямоугольных импульсов (например, «заваливанию» фронтов), чем внутри компьютера. Поэтому для надежного распознавания импульсов на приемном конце линии связи при передаче данных внутри и вне компьютера не всегда можно использовать одни и те же скорости и способы кодирования. Например, медленное нарастание фронта импульса из-за высокой емкостной нагрузки линии требует, чтобы импульсы передавались с меньшей скоростью (чтобы передний и задний фронты соседних импульсов не перекрывались, и импульс успел «дорасти» до требуемого уровня).

В вычислительных сетях применяют как потенциальное, так и импульсное кодирование дискретных данных, а также специфический способ представления данных, который никогда не используется внутри компьютера, — **модуляцию** (рис. 2.8). При модуляции дискретная информация представляется синусоидальным сигналом той частоты, которую хорошо передает имеющаяся линия связи.

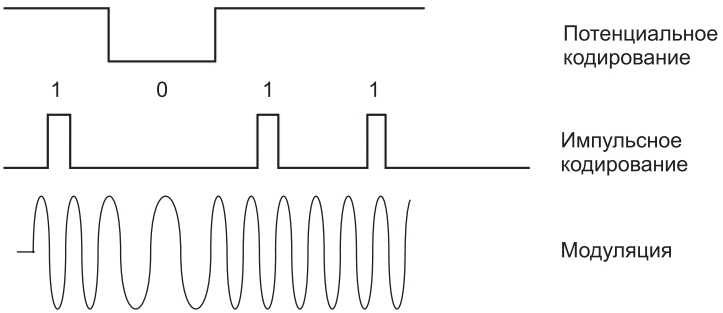


Рис. 2.8. Примеры представления дискретной информации

Потенциальное и импульсное кодирование применяется на каналах *высокого качества*, а модуляция на основе синусоидальных сигналов предпочтительнее в том случае, когда канал вносит сильные искажения в передаваемые сигналы. Например, модуляция используется в глобальных сетях при передаче данных через аналоговые телефонные каналы связи, которые были разработаны для передачи голоса в аналоговой форме и поэтому плохо подходят для непосредственной передачи импульсов.

На способ передачи сигналов влияет и *количество проводов* в линиях связи между компьютерами. Для снижения стоимости линий связи в сетях обычно стремятся к сокращению количества проводов и из-за этого передают все биты одного байта или даже нескольких

байтов не параллельно, как это делается внутри компьютера, а последовательно (побитно), для чего достаточно всего одной пары проводов.

Еще одной проблемой, которую нужно решать при передаче сигналов, является проблема взаимной **синхронизации** передатчика одного компьютера с приемником другого. При организации взаимодействия модулей внутри компьютера эта проблема решается очень просто, так как в этом случае все модули синхронизируются от общего тактового генератора. Проблема синхронизации при связи компьютеров может решаться разными способами — как путем обмена специальными тактовыми синхроимпульсами по отдельной линии, так и путем периодической синхронизации заранее обусловленными кодами или импульсами характерной формы, отличающейся от формы импульсов данных.

Несмотря на предпринимаемые меры (выбор соответствующей скорости обмена данными, линий связи с определенными характеристиками, способа синхронизации приемника и передатчика), вероятность искажения некоторых битов передаваемых данных сохраняется. Для повышения надежности передачи данных между компьютерами, как правило, используется стандартный прием — подсчет **контрольной суммы** и передача полученного значения по линиям связи после каждого байта или после некоторого блока байтов. Часто в протокол обмена данными включается как обязательный элемент **сигнал-квитанция**, который подтверждает правильность приема данных и посылается от получателя отправителю.

Характеристики физических каналов

Существует большое количество характеристик, связанных с передачей трафика через физические каналы. С теми из них, которые будут необходимы нам уже в ближайшее время, мы коротко познакомимся сейчас, а позже изучим их и некоторые другие сетевые характеристики более детально.

- **Предложенная нагрузка** — это поток данных, поступающий от приложения пользователя на вход сети, которая всегда готова принять данные. Предложенную нагрузку можно характеризовать **скоростью генерации данных** в битах в секунду. Эта характеристика описывает интенсивность работы источника информации и абстрагируется от свойств физических каналов.
- **Пропускная способность**[1], называемая также **емкостью канала связи** (capacity), представляет собой *максимально возможную* скорость передачи информации по данному каналу. Эта характеристика никак не связана с предложенной нагрузкой. Она отражает скоростные возможности сети, определяемые параметрами *физической среды передачи*, а также особенностями *выбранного способа передачи* дискретной информации в этой среде. Например, пропускная способность канала связи в сети Ethernet на оптическом волокне равна 10 Мбит/с. Эта скорость является предельно возможной для сочетания технологии Ethernet и оптического волокна. Однако для того же самого оптического волокна можно разработать другую технологию передачи, отличающуюся способом кодирования данных, тактовой частотой и другими параметрами, которая будет иметь другую пропускную способность. Так, технология Fast Ethernet обеспечивает переда-

[1] Читайте более подробно о пропускной способности в разделе «Характеристики линий связи» главы 6.

чу данных по тому же оптическому волокну с максимальной скоростью 100 Мбит/с, а технология Gigabit Ethernet — 1000 Мбит/с.

Для анализа и настройки сети очень полезно знать данные о пропускной способности *отдельных элементов сети*. Из-за последовательного характера передачи данных различными элементами сети общая пропускная способность любого составного пути в сети будет равна *минимальной* из пропускных способностей составляющих элементов маршрута. Для повышения пропускной способности составного пути необходимо в первую очередь обратить внимание на самые медленные элементы, называемые **узкими местами** (bottleneck).

- **Скорость передачи данных**[1] (information rate, или throughput, оба английских термина используются равноправно) — это *фактическая* скорость потока данных, прошедшего через сеть или некоторые ее фрагменты. Скорость информационного потока, называемая также скоростью передачи данных, определяется как частное от деления объема данных, переданных за некоторый интервал времени, на величину этого интервала. Из определения следует, что эта характеристика всегда является усредненной. Она отражает как скорость поступления данных в сеть — предложенную нагрузку, так и скоростные свойства физических каналов — пропускную способность сети. Скорость передачи данных может быть ниже предложенной нагрузки, так как данные в сети могут искажаться или теряться, или на некоторых участках выше — когда предварительно *буферизованные* данные передаются скоростным каналом. Но скорость передачи данных по некоторому каналу никогда не может превысить его пропускной способности.

Еще одна группа характеристик канала связи касается возможности передачи информации по каналу в одну или обе стороны.

При взаимодействии двух компьютеров обычно требуется передавать информацию в обоих направлениях, от компьютера *A* к компьютеру *B* и обратно. Даже в том случае, когда пользователю кажется, что он только получает информацию (например, загружает музыкальный файл из интернета) или только ее передает (отправляет электронное письмо), обмен информацией идет в двух направлениях. Просто существует основной поток данных, которые интересуют пользователя, и вспомогательный поток противоположного направления, который образуют квитанции о получении этих данных.

Физические каналы связи делятся на несколько типов в зависимости от того, могут они передавать информацию в обоих направлениях или нет.

- **Дуплексный канал** обеспечивает одновременную передачу информации в обоих направлениях. Дуплексный канал может состоять из двух физических сред, каждая из которых используется для передачи информации только в одном направлении. Возможен вариант, когда одна среда служит для одновременной передачи встречных потоков. При этом применяют дополнительные методы выделения каждого потока из суммарного сигнала.
- **Полудуплексный канал** также обеспечивает передачу информации в обоих направлениях, но не одновременно, а по очереди. То есть в течение определенного периода времени информация передается в одном направлении, а в течение следующего периода — в обратном.

---

[1] Читайте более подробно о скорости передачи данных в разделе «Характеристики скорости передачи» главы 5.

- **Симплексный канал** позволяет передавать информацию только в одном направлении. Часто дуплексный канал состоит из двух симплексных каналов.

Подробно вопросы физической передачи дискретных данных обсуждаются в части II этой книги.

Проблемы связи нескольких компьютеров

До сих пор мы рассматривали вырожденную сеть, состоящую всего из двух машин. При объединении в сеть большего числа компьютеров возникает целый комплекс новых проблем.

Топология физических связей

Объединяя в сеть несколько (больше двух) компьютеров, необходимо решить, каким образом соединить их друг с другом, другими словами, выбрать конфигурацию физических связей, или их топологию.

Под **топологией сети** понимается конфигурация графа, вершинам которого соответствуют конечные узлы сети (например, компьютеры) и коммуникационное оборудование (например, маршрутизаторы), а ребрам — физические или информационные связи между вершинами.

Число возможных вариантов конфигурации резко возрастает при увеличении числа связываемых устройств. Так, если три компьютера мы можем связать двумя способами (рис. 2.9, *а*), то для четырех можно предложить уже шесть топологически разных конфигураций (при условии неразличимости компьютеров, рис. 2.9, *б*).

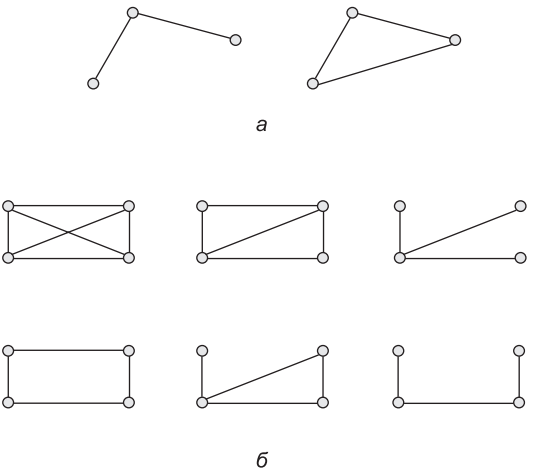


Рис. 2.9. Варианты связи компьютеров

Мы можем соединять каждый компьютер с каждым или же связывать их последовательно, предполагая, что они будут общаться, передавая сообщения друг другу «транзитом». Транзитные узлы должны быть оснащены специальными средствами, позволяющими им

выполнять эту специфическую посредническую операцию. В качестве транзитного узла может выступать как универсальный компьютер, так и специализированное устройство.

От выбора топологии связей существенно зависят характеристики сети. Например, наличие между узлами нескольких путей повышает надежность сети и делает возможным распределение загрузки между отдельными каналами. Простота присоединения новых узлов, свойственная некоторым топологиям, делает сеть легко *расширяемой*. Экономические соображения часто приводят к выбору топологий, для которых характерна минимальная суммарная длина линий связи.

Среди множества возможных конфигураций различают полносвязные и неполносвязные.

Полносвязная топология соответствует сети, в которой каждый компьютер непосредственно связан со всеми остальными (рис. 2.10, *а*). Несмотря на логическую простоту, этот вариант оказывается на практике громоздким и неэффективным. Действительно, в таком случае каждый компьютер в сети должен иметь большое количество коммуникационных портов, достаточное для связи с каждым из остальных компьютеров сети. Для каждой пары компьютеров должна быть выделена отдельная физическая линия связи (в некоторых случаях даже две, если невозможно использование этой линии для двусторонней передачи). Полносвязные топологии в крупных сетях применяются редко, так как для связи N узлов требуется $N(N-1)/2$ физических дуплексных линий связей, то есть имеет место квадратичная зависимость от числа узлов. Обычно этот вид топологии используется в многомашинных комплексах или в сетях, объединяющих небольшое количество компьютеров.

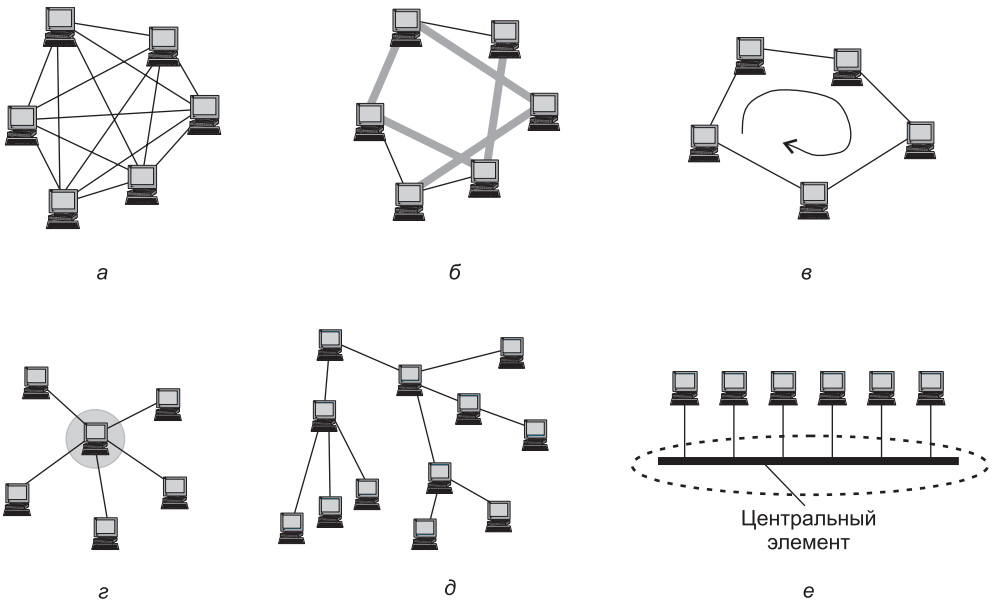


Рис. 2.10. Типовые топологии сетей

Все другие варианты основаны на **неполносвязных топологиях**, когда для обмена данными между двумя компьютерами может потребоваться транзитная передача данных через другие узлы сети.

Ячеистая топология (mesh) получается из полносвязной путем удаления некоторых связей (рис. 2.10, *б*). Ячеистая топология допускает соединение большого количества компьютеров и характерна, как правило, для крупных сетей.

В сетях с **кольцевой топологией** (рис. 2.10, *в*) данные передаются по кольцу от одного компьютера к другому. Главным достоинством кольца является то, что оно по своей природе обеспечивает резервирование связей. Действительно, любая пара узлов соединена здесь двумя путями — по часовой стрелке и против нее. Кроме того, кольцо представляет собой очень удобную конфигурацию для организации обратной связи — данные, сделав полный оборот, возвращаются к узлу-источнику. Поэтому источник может контролировать процесс доставки данных адресату. Часто это свойство кольца используется для тестирования связности сети и поиска узла, работающего некорректно. В то же время в сетях с кольцевой топологией необходимо принимать специальные меры, чтобы в случае выхода из строя или отключения какого-либо компьютера не прерывался канал связи между остальными узлами кольца.

Звездообразная топология (рис. 2.10, *г*) образуется в случае, когда каждый компьютер подключается непосредственно к общему центральному устройству, называемому **концентратором**[1]. В функции концентратора входит направление передаваемой компьютером информации одному или всем остальным компьютерам сети. В качестве концентратора может выступать как универсальный компьютер, так и специализированное устройство. К недостаткам звездообразной топологии относится более высокая стоимость сетевого оборудования из-за необходимости приобретения специализированного центрального устройства. Кроме того, возможности по наращиванию количества узлов в сети ограничиваются количеством портов концентратора.

Иногда имеет смысл строить сеть с использованием нескольких концентраторов, иерархически соединенных между собой звездообразными связями (рис. 2.10, *д*). Получаемую в результате структуру называют **иерархической звездой** или **деревом**. В настоящее время дерево является самой распространенной топологией связей как в локальных, так и глобальных сетях.

Особым частным случаем звезды является **общая шина** (рис. 2.10, *е*). Здесь в качестве центрального элемента выступает пассивный кабель, к которому по схеме «монтажного ИЛИ» подключается несколько компьютеров (такую же топологию имеют многие сети, использующие беспроводную связь, — роль общей шины здесь играет общая радиосреда). Передаваемая информация распространяется по кабелю и доступна одновременно всем компьютерам, присоединенным к этому кабелю. Основными преимуществами такой схемы являются ее дешевизна и простота присоединения новых узлов к сети, а недостатками — низкая надежность (любой дефект кабеля полностью парализует всю сеть) и невысокая производительность (в каждый момент времени только один компьютер может передавать данные по сети, поэтому пропускная способность делится здесь между всеми узлами сети).

В то время как небольшие сети, как правило, имеют типовую топологию — звезда, кольцо или общая шина, — для крупных сетей характерно наличие произвольных связей между компьютерами. В таких сетях можно выделить отдельные произвольно связанные фрагмен-

---

[1] В данном случае термин «концентратор» используется в широком смысле, обозначая любое многовходовое устройство, способное служить центральным элементом — например, коммутатор или маршрутизатор.

ты (подсети), имеющие типовую топологию, поэтому их называют сетями со **смешанной топологией** (рис. 2.11).

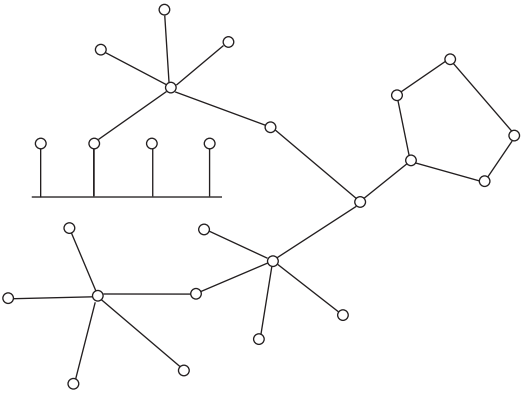


Рис. 2.11. Смешанная топология

Адресация узлов сети

Еще одной новой проблемой, которую нужно учитывать при объединении трех и более компьютеров, является проблема их адресации, точнее, адресации их сетевых интерфейсов[1]. Один компьютер может иметь несколько сетевых интерфейсов. Например, для создания полносвязной структуры из N компьютеров необходимо, чтобы у каждого из них имелся $N - 1$ интерфейс.

По количеству адресуемых интерфейсов адреса можно классифицировать следующим образом:

- **уникальный адрес** (unicast) используется для идентификации отдельных интерфейсов;
- **групповой адрес** (multicast) идентифицирует сразу несколько интерфейсов, поэтому данные, помеченные групповым адресом, доставляются каждому из узлов, входящих в группу;
- данные, направленные по **широковещательному адресу** (broadcast), должны быть доставлены всем узлам сети;
- **адрес произвольной рассылки** (anycast), так же как и групповой адрес, задает группу адресов, однако данные, посланные по этому адресу, доставляются не всем узлам данной группы, а только одному из них. Выбор этого узла осуществляется в соответствии с некоторыми правилами предпочтения.

Адреса могут быть **числовыми** (например, 129.26.255.255 или 81.1a.ff.ff) и **символьными** (site.domen.ru, willi-winki).

Символьные адреса (имена) предназначены для запоминания людьми и поэтому обычно несут смысловую нагрузку. Для работы в больших сетях символьное имя может иметь иерархическую структуру, например ftp-arch1.ucl.ac.uk. Этот адрес говорит о том, что данный компьютер поддерживает ftp-архив в сети одного из колледжей Лондонского

[1] Иногда вместо точного выражения «адрес сетевого интерфейса» мы будем использовать упрощенное — «адрес узла сети».

университета (University College London — ucl) и эта сеть относится к академической ветви (ac) интернета Великобритании (United Kingdom — uk). При работе в пределах сети Лондонского университета такое длинное символьное имя явно избыточно, и вместо него можно пользоваться кратким символьным именем ftp-arch1. Хотя символьные имена удобны для людей, из-за переменного формата и потенциально большой длины их передача по сети не экономична.

Множество всех адресов, которые являются допустимыми в рамках некоторой схемы адресации, называется **адресным пространством**. Адресное пространство может иметь плоскую (линейную) или иерархическую организацию.

При **плоской организации** множество адресов никак не структурировано. Примером плоского числового адреса является **MAC-адрес**, предназначенный для однозначной идентификации сетевых интерфейсов в локальных сетях. Такой адрес обычно используется только аппаратурой, поэтому его стараются сделать по возможности компактным и записывают в виде двоичного или шестнадцатеричного числа, например 0081005e24a8. При задании MAC-адресов не требуется выполнять никакой ручной работы, так как они обычно встраиваются в аппаратуру компанией-изготовителем, поэтому их называют также **аппаратными адресами** (hardware address). Использование плоских адресов является жестким решением — при замене аппаратуры, например сетевого адаптера, изменяется и адрес сетевого интерфейса компьютера.

При **иерархической** организации адресное пространство структурируется в виде вложенных друг в друга подгрупп, которые, последовательно сужая адресуемую область, в конце концов определяют отдельный сетевой интерфейс. Например, в трехуровневой структуре адресного пространства адрес конечного узла может задаваться тремя составляющими:

- идентификатором группы (K), в которую входит данный узел;
- идентификатором подгруппы (L);
- идентификатором узла (n), однозначно определяющим его в подгруппе.

Иерархическая адресация во многих случаях оказывается более рациональной, чем плоская. В больших сетях, состоящих из многих тысяч узлов, использование плоских адресов приводит к большим издержкам — конечным узлам и коммуникационному оборудованию приходится оперировать таблицами адресов, состоящими из тысяч записей. В противоположность этому иерархическая система адресации позволяет при перемещении данных до определенного момента пользоваться только старшей составляющей адреса (например, идентификатором группы K), затем (для дальнейшей локализации адресата) задействовать следующую по старшинству часть (L) и в конечном счете — младшую часть (n).

Типичными представителями иерархических числовых адресов являются сетевые IP-адреса. В них поддерживается двухуровневая иерархия, адрес делится на старшую часть — номер сети и младшую — номер узла. Такое деление позволяет передавать сообщения между сетями только на основании номера сети, а номер узла требуется уже после доставки сообщения в нужную сеть — точно так же, как название улицы используется почтальоном только после того, как письмо доставлено в нужный город.

На практике обычно применяют сразу несколько схем адресации, так что сетевой интерфейс компьютера может одновременно иметь несколько адресов-имен. Каждый адрес задействуется в той ситуации, когда соответствующий вид адресации наиболее удобен.

А для преобразования адресов из одного вида в другой используются специальные вспомогательные протоколы, которые называют **протоколами разрешения адресов**.

Пользователи адресуют компьютеры иерархическими символьными именами, которые автоматически заменяются в сообщениях, передаваемых по сети, иерархическими числовыми адресами. С помощью этих числовых адресов сообщения доставляются из одной сети в другую, а после доставки сообщения в сеть назначения вместо иерархического числового адреса используется плоский аппаратный адрес компьютера. Проблема установления соответствия между адресами различных типов может решаться как централизованными, так и распределенными средствами.

При *централизованном подходе* в сети выделяется один или несколько компьютеров (серверов имен), в которых хранится таблица соответствия имен различных типов — например, символьных имен и числовых адресов. Все остальные компьютеры обращаются к серверу имен с запросами, чтобы по символьному имени найти числовой номер необходимого компьютера.

При *распределенном подходе* каждый компьютер сам хранит все назначенные ему адреса разного типа. Тогда компьютер, которому необходимо определить по известному иерархическому числовому адресу некоторого компьютера его плоский аппаратный адрес, посылает в сеть широковещательный запрос. Все компьютеры сети сравнивают содержащийся в запросе адрес с собственным. Тот компьютер, у которого обнаружилось совпадение, посылает ответ, содержащий искомый аппаратный адрес. Такая схема использована в **протоколе разрешения адресов** (Address Resolution Protocol, ARP) стека TCP/IP.

Достоинство распределенного подхода состоит в том, что он позволяет отказаться от выделения специального компьютера в качестве сервера имен, который к тому же часто требует ручного задания таблицы соответствия адресов. Недостатком его является необходимость широковещательных сообщений, перегружающих сеть. Именно поэтому распределенный подход используется в небольших сетях, а централизованный — в больших.

До сих пор мы говорили об адресах сетевых интерфейсов, компьютеров и коммуникационных устройств, однако конечной целью данных, пересылаемых по сети, являются не сетевые интерфейсы или компьютеры, а выполняемые на этих устройствах программы — процессы. Поэтому в адресе назначения наряду с информацией, идентифицирующей интерфейс устройства, должен указываться адрес процесса, которому предназначены посылаемые по сети данные. Очевидно, что достаточно обеспечить уникальность адреса процесса в пределах компьютера. Примером адресов процессов являются *номера портов TCP и UDP*, используемые в стеке TCP/IP.

Коммутация

Пусть компьютеры физически связаны между собой в соответствии с некоторой топологией, выбрана система адресации. Остается нерешенным вопрос: каким образом передавать данные между конечными узлами? Особую сложность приобретает эта задача для неполносвязной топологии сети, когда обмен данными между произвольной парой конечных узлов (пользователей) должен идти в общем случае через транзитные узлы.

Соединение конечных узлов через сеть транзитных узлов называют **коммутацией**. Последовательность узлов, лежащих на пути от отправителя к получателю, образует **маршрут**.

Например, в сети, показанной на рис. 2.12, узлы 2 и 4, непосредственно между собой не связанные, вынуждены передавать данные через транзитные узлы, в качестве которых могут выступить, например, узлы 1 и 5. Узел 1 должен выполнить передачу данных между своими интерфейсами *A* и *B*, а узел 5 — между интерфейсами *F* и *B*. В данном случае маршрутом является последовательность: 2-1-5-4, где 2 — узел-отправитель, 1 и 5 — транзитные узлы, 4 — узел-получатель.

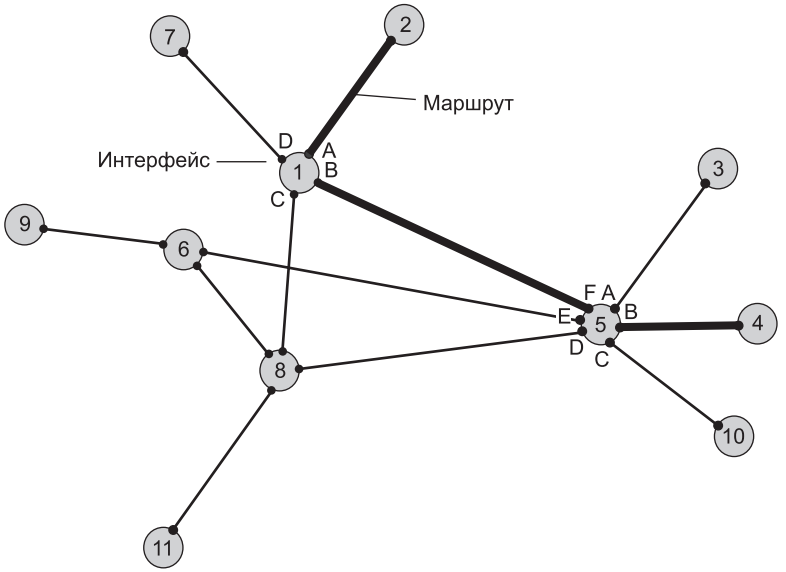


Рис. 2.12. Коммутация абонентов через сеть транзитных узлов

Обобщенная задача коммутации

В самом общем виде задача коммутации может быть представлена в виде следующих взаимосвязанных частных задач:

1. **Определение** информационных потоков, для которых требуется прокладывать маршруты.
2. **Маршрутизация** потоков (routing) — прокладка маршрутов.
3. **Продвижение** потоков (forwarding), то есть распознавание потоков и их локальная коммутация на каждом транзитном узле.
4. **Мультиплексирование** и **демультиплексирование** потоков.

Определение информационных потоков

Понятно, что через один транзитный узел может проходить несколько маршрутов, например, через узел 5 (см. рис. 2.12) проходят как минимум все данные, направляемые узлом 4 каждому из остальных узлов, а также все данные, поступающие в узлы 3, 4 и 10. Транзитный узел должен уметь *распознавать* поступающие на него потоки данных, чтобы обеспе-

чивать передачу каждого из них именно на тот свой интерфейс, который ведет к нужному узлу, и, возможно, чтобы выбрать специфический для данного потока способ его обработки.

Информационным потоком, или потоком данных, называют непрерывную последовательность данных, объединенных набором общих признаков, выделяющих эти данные из общего сетевого трафика.

Например, как поток можно определить все данные, поступающие от одного компьютера; объединяющим признаком в данном случае служит адрес источника. Эти же данные можно представить как совокупность нескольких **подпотоков**, каждый из которых в качестве дифференцирующего признака имеет адрес назначения. Наконец, каждый из этих подпотоков, в свою очередь, можно разделить на более мелкие подпотоки, порожденные разными сетевыми приложениями — электронной почтой, программой копирования файлов, веб-сервером. Данные, образующие поток, могут быть представлены в виде последовательности различных информационных единиц данных — пакетов, кадров, ячеек.

Очевидно, что при коммутации в качестве *обязательного признака* выступает **адрес назначения** данных. На основании этого признака весь поток входящих в транзитный узел данных разделяется на подпотоки, каждый из которых передается на интерфейс, соответствующий тому или иному маршруту продвижения данных.

Адреса источника и назначения определяют поток для пары соответствующих конечных узлов. Однако часто бывает полезно представить этот поток в виде нескольких подпотоков, причем для каждого из них может быть проложен свой особый маршрут. Рассмотрим пример, когда на одной и той же паре конечных узлов выполняется несколько взаимодействующих по сети приложений, каждое из которых предъявляет к сети свои особые требования. В таком случае выбор маршрута должен осуществляться с учетом характера передаваемых данных, например, для файлового сервера важно, чтобы передаваемые им большие объемы данных направлялись по каналам, обладающим высокой пропускной способностью, а для программной системы управления, которая посылает в сеть короткие сообщения, требующие обязательной и немедленной отработки, при выборе маршрута более важна надежность линии связи и минимальный уровень задержек на маршруте. Кроме того, даже для данных, предъявляющих к сети одинаковые требования, может прокладываться несколько маршрутов, чтобы за счет распараллеливания ускорить передачу данных.

Возможна и обратная по отношению к выделению подпотоков операция — *агрегирование потоков*. Обычно она выполняется на магистралях сетей, которые передают очень большое количество индивидуальных потоков. Агрегирование потоков, имеющих общую часть маршрута через сеть, позволяет уменьшить количество хранимой промежуточными узлами сети информации, так как агрегированные потоки описываются в них как одно целое. В результате снижается нагрузка на промежуточные узлы сети и повышается их быстродействие.

Признаки потока могут иметь *глобальное* или *локальное* значение — в первом случае они однозначно определяют поток в пределах всей сети, а во втором — в пределах одного транзитного узла. Пара идентифицирующих поток адресов конечных узлов — это пример глобального признака. Примером признака, локально определяющего поток в пределах устройства, может служить номер (идентификатор) интерфейса данного устройства, на который поступили данные. Например, возвращаясь к рис. 2.12, узел 1 может быть настроен так, чтобы передавать на интерфейс *B* все данные, поступившие с интерфейса *A*, а на интерфейс *C* — данные, поступившие с интерфейса *D*. Такое правило позволяет отделить поток

данных узла 2 от потока данных узла 7 и направлять их для транзитной передачи через разные узлы сети, в данном случае поток узла 2 — через узел 5, а поток узла 7 — через узел 8.

Метка потока — это особый тип признака. Она представляет собой некоторое число, которое несут все данные потока. **Глобальная метка** назначается данным потока и не меняет своего значения на всем протяжении его пути следования от узла источника до узла назначения, таким образом, она уникально определяет поток в пределах сети. В некоторых технологиях используются **локальные метки** потока, динамически меняющие свое значение при передаче данных от одного узла к другому.

Таким образом , распознавание потоков во время коммутации происходит на основании признаков, в качестве которых, помимо обязательного адреса назначения данных, могут выступать и другие признаки — такие, например, как идентификаторы приложений. Набор признаков, определяющий поток, называется **дескриптором потока**.

Маршрутизация

Задача маршрутизации, в свою очередь, включает в себя две подзадачи:

- определение маршрутов;
- оповещение сети о найденных маршрутах.

Определить маршрут означает выбрать последовательность транзитных узлов и их интерфейсов, через которые надо передавать данные, чтобы доставить их адресату. Определение маршрута — сложная задача, особенно когда конфигурация сети такова, что между парой взаимодействующих сетевых интерфейсов существует множество путей. Чаще всего выбор останавливают на одном *оптимальном* по некоторому критерию маршруте. В качестве критериев оптимальности могут выступать, например, пропускная способность и загруженность каналов связи; задержки, вносимые каналами; количество промежуточных транзитных узлов; надежность каналов и транзитных узлов.

Формально выбор маршрута можно описать как задачу минимизации **длины маршрута**[1] на заданном графе. На рис. 2.13 показан граф, вершины которого представляют узлы сети, а ребра — линии связи между соседними узлами. Каждой линии связи (или просто «связи») приписано число — **длина линии связи**, здесь *C*1 — длина связи *A*-*B*, *C*2 — длина

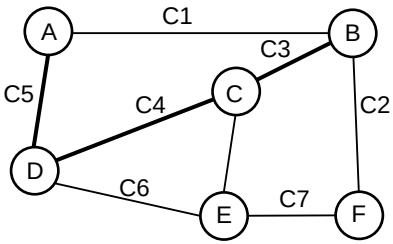


Рис. 2.13. Задача поиска маршрута минимальной длины

[1] Когда речь идет о задаче поиска оптимального маршрута, часто вместо терминов «длина маршрута»/«длина линии связи» используют равнозначные в данном контексте термины «стоимость маршрута» «стоимость линии связи».

связи *B-F*, и т. д. Любую пару узлов сети в общем случае можно связать несколькими путями, наилучшим из которых считается тот, который имеет минимальную длину. В данном контексте «длина» является абстрактной характеристикой линии связи, которая может измеряться в разных **метриках**: от обычных единиц измерения длины (метры, километры и т. д.) или денежных единиц (стоимость аренды линии связи) до специфических сетевых характеристик каналов связи — пропускной способности, надежности, вносимых задержек и любых комбинаций этих метрик.

Метрики должны выбираться таким образом, чтобы выполнялось *свойство аддитивности*, то есть длина любого пути на графе всегда была равна сумме длин линий связи, из которых складывается данный путь. Например, чтобы длина маршрута *A-D-C-B*, показанного на рисунке, была равна *C5+C4+C3*. Такое свойство очевидно выполняется, если в качестве метрики выбраны единицы измерения физического расстояния, но оно очевидно не выполняется, если длина измеряется, например, единицами пропускной способности.

Длина маршрута может быть измерена в количестве **хопов** (прыжков) между маршрутизаторами. Если каждой линии связи приписать значение длины, равное 1, то длина маршрута будет равна количеству маршрутизаторов, которые нужно преодолеть пакету до сети назначения. Эта метрика часто применяется в протоколах маршрутизации для выбора оптимального маршрута, однако она работает не всегда. На рис. 2.14 для передачи трафика между конечными узлами *A* и *C* существуют два альтернативных маршрута: *A*-1-2-3-*C* и *A*-1-3-*C*. Если мы учитываем только количество хопов, то выбор очевиден — маршрут *A*-1-3-*C*, который имеет меньше транзитных узлов. Однако, возможно, наш выбор был не самым лучшим. На рисунке показано, что каналы 1-2 и 2-3 обладают пропускной способностью 100 Гбит/с каждый, а канал 1-3 — только 10 Гбит/с. Чтобы информация передавалась по сети с максимально возможной скоростью, следовало бы выбрать маршрут *A*-1-2-3-*C*, хотя он и проходит через большее количество промежуточных узлов. То есть можно сказать, что маршрут *A*-1-2-3-*C* в данном случае оказывается «более коротким».

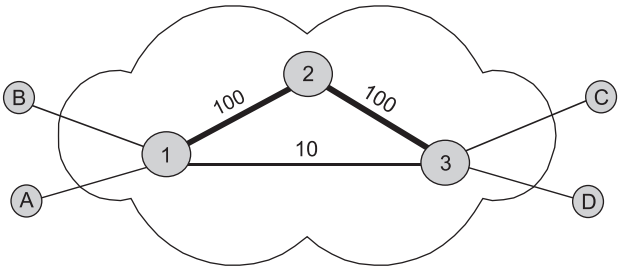


Рис. 2.14. Метрика, учитывающая пропускную способность

Для построения метрики, учитывающей пропускную способность, часто применяют следующий прием: длину каждого канала участка характеризуют величиной, обратной его пропускной способности. Чтобы оперировать целыми числами, выбирают некоторую константу, заведомо бóльшую, чем пропускные способности каналов в сети. Например, если мы в качестве такой константы выберем 100 Гбит/с, то длина каждого из каналов 1-2 и 2-3 будет равна 1, а длина канала 1-3 составит 10. Длина маршрута равна сумме длин составляющих его каналов, поэтому путь 1-2-3, длина которого равна 2, «короче» альтернативного пути 1-3, имеющего длину 10.

Маршрут может определяться эмпирически («вручную») администратором сети на основании различных, часто не формализуемых соображений. Среди побудительных мотивов выбора пути могут быть: особые требования к сети со стороны различных типов приложений, решение передавать трафик через сеть определенного поставщика услуг, предположения о пиковых нагрузках на некоторые каналы сети, соображения безопасности.

Однако эмпирический подход к определению маршрутов малопригоден для большой сети со сложной топологией. В этом случае используются автоматические методы определения маршрутов. Они могут быть *централизованными*, когда маршруты вычисляются на компьютере, в общем случае не связанном с сетью, либо *распределенными* — в этом случае все узлы сети оснащаются специальными программными средствами, которые организуют взаимный обмен служебными сообщениями, позволяющий им коллективно определить рациональные маршруты.

После того как маршруты определены (вручную или автоматически), информация о них должна быть доведена до всех узлов сети. Для этого на каждом узле создается специальная таблица, называемая **таблицей коммутации/маршрутизации/продвижения**. Информация о каждом маршруте представлена в таблицах в распределенном виде: вместо описания маршрута в виде полного перечня транзитных узлов от узла отправления до узла получателя в таблице помещается лишь фрагмент маршрута, касающийся данного узла. Фрагмент оформлен в виде записи, упрощенно представленной в табл. 2.1, и несет примерно такую информацию: «Каждый раз, когда в устройство поступят данные потока *n* с признаками {адрес назначения DA, адрес источника SA, тип приложения A}, их следует передать для дальнейшего продвижения на интерфейс *if*1»[1].

В результате обработки информации обо всех маршрутах на каждом узле формируется собственная таблица коммутации, содержащая столько записей, сколько различных маршрутов проходит через этот узел. В совокупности таблицы продвижения содержат в преобразованном виде информацию обо всех маршрутах.

Таблица 2.1. Фрагмент таблицы коммутации

Признаки потока	Направление передачи данных (номер интерфейса и/или адрес следующего узла)
n ={DA, SA, A}	*if*1

Формирование таблиц коммутации, как и нахождение маршрутов, может осуществляться централизованно или распределенным способом, вручную или автоматически. Администратор сети может зафиксировать маршрут, выполнив в ручном режиме конфигурирование устройства, например, жестко скоммутировав на длительное время определенные пары входных и выходных интерфейсов (как работали «телефонные барышни» на первых коммутаторах). Он может также по собственной инициативе внести запись о маршруте в таблицу коммутации.

[1] В том случае, когда к одному интерфейсу может быть подключено несколько узлов, в записи указывается также адрес следующего транзитного узла.

Однако поскольку топология сети и состав информационных потоков могут меняться (отказы узлов или появление новых промежуточных узлов, изменение адресов или определение новых потоков), требуется гибкое решение задач определения маршрутов и формирования таблиц продвижения с учетом текущего состояния сети. В таких случаях задачи прокладки маршрутов, как правило, не могут быть решены без участия достаточно сложных программных и аппаратных средств. Наиболее часто эту работу выполняют сами коммутирующие устройства – **маршрутизаторы**.

Продвижение данных

Когда маршруты определены и записи о них сделаны в таблицах всех узлов, может быть выполнена передача данных между абонентами (коммутация абонентов). Эта функция коммутационных устройств называется **продвижением** данных.

Для каждой пары абонентов эта операция может быть представлена несколькими (по числу транзитных узлов) *локальными* операциями продвижения. Прежде всего отправитель должен выставить данные на тот свой интерфейс, с которого начинается найденный маршрут, а все транзитные узлы должны соответствующим образом выполнить «переброску» данных с одного своего интерфейса на другой, другими словами, выполнить **коммутацию интерфейсов**. Устройство, функциональным назначением которого является коммутация, называется **коммутатором**. На рис. 2.15 показан коммутатор, который переключает информационные потоки между четырьмя своими интерфейсами.

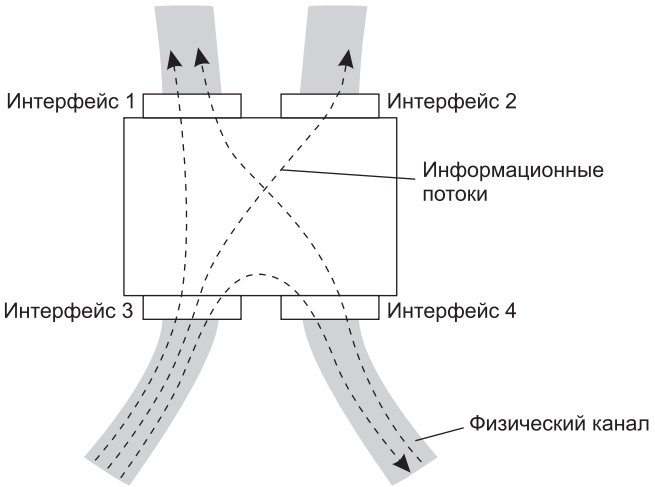


Рис. 2.15. Коммутатор

Интерфейс коммутатора (называемый также **портом**) является физическим модулем, состоящим из приемника и передатчика. В том случае, когда передатчик и приемник работают на дуплексный канал связи, они работают независимо друг от друга, обеспечивая одновременную передачу данных в обоих направлениях. Иногда приемную часть интерфейса называют **входным интерфейсом**, а выходную часть – **выходным интерфейсом**.

Функция продвижения включает в себя следующие основные действия:

- *Получение данных* на входной интерфейс устройства.
- *Распознавание потока* — из поступивших данных коммутатор извлекает признаки потока, на основании которых в таблице различаются маршруты, например адрес назначения.
- *Просмотр таблицы* коммутации с целью поиска записи о потоке с найденным значением признака.
- *Извлечение из записи таблицы коммутации информации* о том, на какой интерфейс данного коммутатора должны быть переданы данные, а также, если необходимо, адрес следующего транзитного коммутатора.
- В случае необходимости выполняется также *модификация поступивших данных*, например, коммутатор может поместить в поле служебной информации адрес следующего коммутатора, изменить контрольную сумму, поставить отметку о времени прохождения через данный коммутатор и др.
- *Передача данных* на заданный в таблице коммутации интерфейс коммутатора.

Задача маршрутизации (определение маршрутов и формирования таблиц продвижения) и задача продвижения данных являются двумя *независимыми задачами*, решаемыми в общем случае в разные моменты времени и разными устройствами. Для устройства, выполняющего продвижение данных, не важно, каким способом были сформированы таблицы маршрутизации — вручную администратором, или вычислены автономным сервером и переданы в готовом виде коммутаторам, или они являются результатом согласованной работы программных модулей, установленных на маршрутизаторах.

Коммутатором может быть как специализированное устройство, так и универсальный компьютер со встроенным программным механизмом коммутации, в этом случае коммутатор называется *программным*. Компьютер может совмещать функции коммутации данных с выполнением своих обычных функций как конечного узла. Однако во многих случаях более рациональным является решение, в соответствии с которым некоторые узлы в сети выделяются *специально* для коммутации. Эти узлы образуют **коммутационную сеть**, к которой подключаются все остальные. На рис. 2.16 показана коммутационная сеть, образованная из узлов 1, 5, 6 и 8, к которой подключаются конечные узлы 2, 3, 4, 7, 9 и 10.

О ТЕРМИНАХ

Термины «коммутация», «таблица коммутации» и «коммутатор» в телекоммуникационных сетях могут трактоваться неоднозначно. Мы уже определили коммутацию как процесс соединения абонентов сети через транзитные узлы. Этим же термином мы обозначаем и соединение интерфейсов в пределах отдельного транзитного узла. Коммутатором в широком смысле называется устройство любого типа, способное выполнять операции переключения потока данных с одного интерфейса на другой. Операция коммутации может выполняться в соответствии с различными правилами и алгоритмами. Некоторые способы коммутации и соответствующие им таблицы и устройства получили специальные названия. Например, в технологии IP для обозначения аналогичных понятий используются термины «маршрутизация», «таблица маршрутизации», «маршрутизатор». В то же время за другими специальными типами коммутации и соответствующими устройствами закрепились те же самые названия, «коммутация», «таблица коммутации» и «коммутатор», применяемые в узком смысле, например, как коммутация и коммутатор в локальной сети Ethernet. Для телефонных сетей, которые появились намного раньше компьютерных, также характерна аналогичная терминология, «коммутатор» является здесь синонимом «телефонной станции».

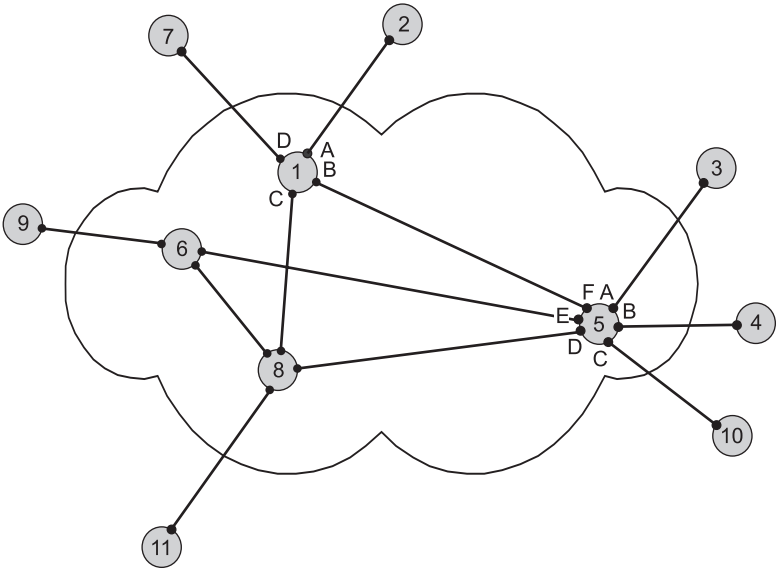


Рис. 2.16. Коммутационная сеть

Мультиплексирование и демультиплексирование

Чтобы определить, на какой интерфейс следует передать поступившие данные, коммутатор должен выяснить, к какому потоку они относятся. Эта задача должна решаться независимо от того, поступает на вход коммутатора только один «чистый» поток или «смешанный» поток, являющийся результатом агрегирования нескольких потоков. В последнем случае к задаче распознавания потоков добавляется задача демультиплексирования.

Демультиплексирование — разделение суммарного потока на несколько составляющих его потоков.

Как правило, операцию коммутации сопровождает также обратная операция мультиплексирования.

Мультиплексирование (**агрегирование**) — образование из нескольких отдельных потоков общего агрегированного потока, который передается по одному физическому каналу связи.

Операции мультиплексирования/демультиплексирования имеют такое же важное значение в любой сети, как и операции коммутации, потому что без них пришлось бы для каждого потока предусматривать отдельный канал, что привело бы к большому количеству параллельных связей в сети и свело бы на нет все преимущества неполносвязной сети.

На рис. 2.17 показан фрагмент сети, состоящий из трех коммутаторов. Коммутатор 1 имеет четыре сетевых интерфейса. На интерфейс 1 поступают данные с двух интерфейсов — 3 и 4. Их надо передать в общий физический канал, то есть выполнить операцию мультиплексирования.

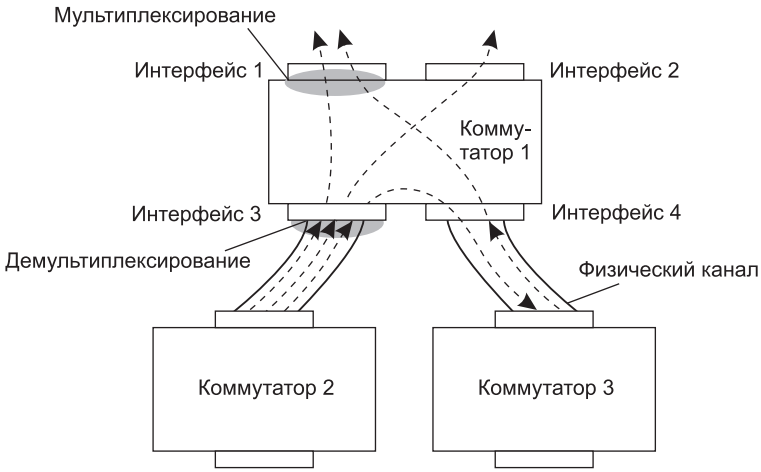


Рис. 2.17. Операции мультиплексирования и демультиплексирования потоков при коммутации

Одним из основных способов мультиплексирования потоков является **разделение времени**. При этом способе каждый поток в определенные интервалы времени (с фиксированным или случайным периодом) получает физический канал в полное свое распоряжение и передает по нему свои данные. Распространено также **частотное разделение** канала, когда каждый поток передает данные в выделенном ему частотном диапазоне.

Технология мультиплексирования должна позволять получателю такого суммарного потока выполнять обратную операцию — разделение (демультиплексирование) данных на слагаемые потоки. На интерфейсе 3 коммутатор выполняет демультиплексирование потока на три составляющих его подпотока. Один из них он передает на интерфейс 1, другой — на интерфейс 2, третий — на интерфейс 4.

Вообще говоря, на одном интерфейсе могут одновременно выполняться обе функции — мультиплексирование и демультиплексирование.

Частный случай коммутатора, у которого все входящие информационные потоки коммутируются на один выходной интерфейс, где они мультиплексируются в один агрегированный поток, называется **мультиплексором**. Коммутатор, который имеет один входной интерфейс и несколько выходных, называется **демультиплексором** (рис. 2.18).

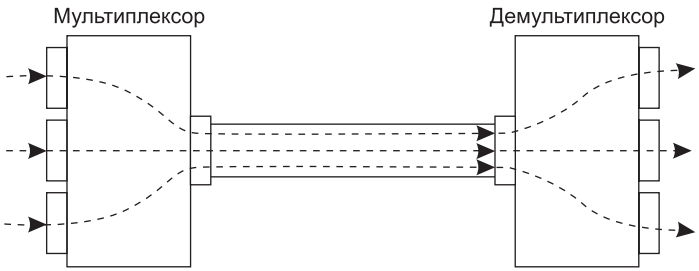


Рис. 2.18. Мультиплексор и демультиплексор

Разделяемая среда передачи данных

Во всех рассмотренных примерах мультиплексирования потоков к каждой линии связи подключались только два интерфейса. В том случае, когда линия связи является дуплексным каналом связи, как это показано на рис. 2.19, каждый из интерфейсов монопольно использует канал связи в направлении «от себя». Это объясняется тем, что дуплексный канал состоит из двух независимых сред передачи данных (подканалов), и так как только передатчик интерфейса является активным устройством, а приемник пассивно ожидает поступления сигналов от него, то и конкуренции подканалов не возникает. Такой режим использования среды передачи данных является в настоящее время основным в компьютерных локальных и глобальных сетях.

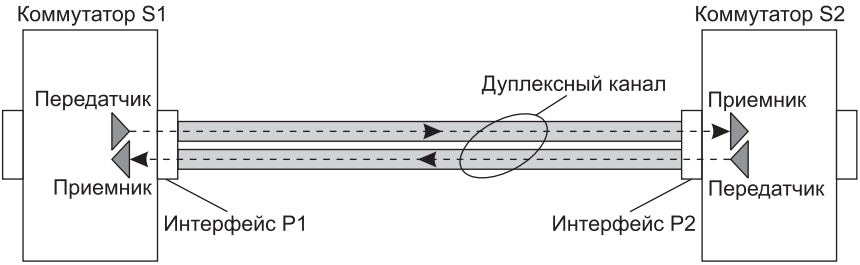


Рис. 2.19. Дуплексный канал — разделяемая среда отсутствует

Однако если в глобальных сетях такой режим использовался всегда, то в локальных сетях до середины 90-х годов преобладал другой режим, основанный на разделяемой среде передачи данных.

Разделяемой средой (shared medium) называется физическая среда передачи данных, к которой непосредственно подключено несколько передатчиков узлов сети. Причем в каждый момент времени только один из передатчиков и приемников какого-либо узла сети получает доступ к разделяемой среде и использует ее для передачи данных приемнику другого узла, подключенному к этой же среде.

В наиболее простом случае эффект разделения среды возникает при соединении двух интерфейсов с помощью одного полудуплексного канала связи, то есть такого канала, который может передавать данные в любом направлении, но только попеременно (рис. 2.20). В этом случае к одной и той же среде передачи данных (например, к коаксиальному кабелю или общей радиосреде) подключены два приемника и два передатчика двух независимых узлов сети.

При подобном применении среды передачи данных возникает новая задача *совместного использования среды независимыми передатчиками* таким образом, чтобы в каждый отдельный момент времени по среде передавались данные только одного передатчика. Другими словами, возникает *необходимость в механизме синхронизации доступа интерфейсов к разделяемой среде*.

Обобщением разделяемой среды является случай, показанный на рис. 2.21, когда к каналу связи подключаются более двух интерфейсов (в приведенном примере — три), при этом применяется топология общей шины.

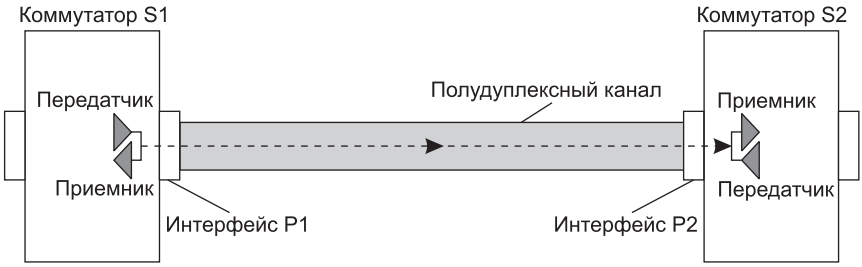


Рис. 2.20. Полудуплексный канал — разделяемая среда

Существуют различные способы решения задачи организации совместного доступа к разделяемым линиям связи. Одни из них подразумевают *централизованный* подход, когда доступом к каналу управляет специальное устройство — **арбитр**. Если мы обратимся к организации работы компьютера, то увидим, что доступ к системной шине компьютера, которую совместно используют внутренние блоки компьютера, управляется централизованно — либо процессором, либо специальным арбитром шины.

Другим способом организации совместного доступа к разделяемым линиям связи является *децентрализованный* доступ. В этом случае решение о предоставлении доступа принимается путем взаимодействия узлов, подключенных к разделяемой среде, между собой.

В глобальных сетях организация совместного доступа к линиям связи имеет свою специфику из-за существенно большего времени распространения сигналов по линиям связи. Здесь процедуры согласования доступа к линии связи могут занимать слишком большой промежуток времени и приводить к значительным потерям производительности сети. Именно по этой причине разделяемая среда в глобальных сетях практически не используется.

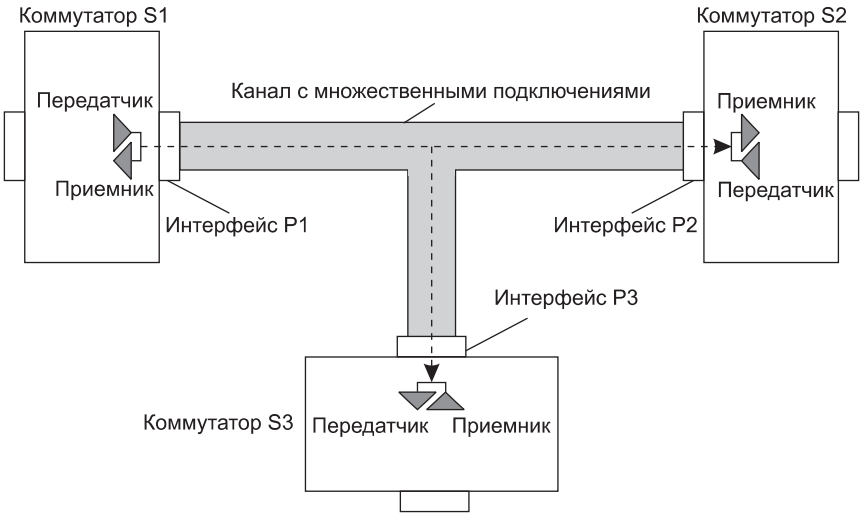


Рис. 2.21. Канал с множественными подключениями — разделяемая среда

Разделяемая среда является частным случаем мультиплексирования потоков. Ее специфика заключается в том, что потоки принадлежат разным узлам сети, при этом они мультиплексируются на физической среде передачи данных. Признаком потока является только тот факт, что пакеты, принадлежащие потоку, были переданы одним и тем же передатчиком одного и того же узла. Другие признаки потока, например адреса источников или назначения пакетов, во внимание не принимаются.

Для локальных сетей разделяемая среда сравнительно долго была основным механизмом использования каналов связи, который применялся во всех технологиях локальных сетей — Ethernet, Token Ring, FDDI. При этом в технологиях локальных сетей применялись децентрализованные методы доступа к среде, не требующие наличия арбитра в сети. Популярность техники разделения среды в локальных сетях объяснялась простотой и экономичностью аппаратных решений.

Сегодня в проводных локальных сетях метод разделения среды практически перестал применяться. Основной причиной отказа от разделяемой среды явилась ее низкая и плохо предсказуемая производительность, а также плохая масштабируемость[1]. Низкая производительность объясняется тем, что пропускная способность канала связи *делится* между всеми компьютерами сети. Например, если локальная сеть Ethernet состоит из 100 компьютеров, а для их связи используются коаксиальный кабель и сетевые адаптеры, работающие на скорости 10 Мбит/с, то в среднем на каждый компьютер приходится только 0,1 Мбит/с пропускной способности. Более точно оценить долю пропускной способности, приходящуюся на какой-либо компьютер сети, трудно, так как эта величина зависит от многих случайных факторов — например, активности других компьютеров. Наверное, к этому моменту читателю уже понятна причина плохой масштабируемости подобной сети — чем больше мы добавляем компьютеров, тем меньшая доля пропускной способности достается каждому компьютеру сети.

Описанные недостатки являются следствием самого принципа разделения среды, поэтому преодолеть их полностью невозможно. Появление в начале 90-х недорогих коммутаторов локальных сетей привело к настоящей революции в этой области, и постепенно коммутаторы вытеснили разделяемую среду полностью.

Сегодня механизм разделения среды используется только в беспроводных локальных сетях, где общая среда — радиоэфир — естественным образом соединяет все конечные узлы, находящиеся в зоне распространения сигнала.

Пример-аналогия

Поясним обобщенную модель коммутации на примере работы традиционной почтовой службы. Аналогом информационного потока компьютерной сети здесь являются почтовые отправления. Основным признаком почтового потока является адрес получателя. Для упрощения будем рассматривать в качестве адреса только страну, например Индия, Норвегия, Россия, Бразилия и т. д. Дополнительным признаком потока может служить особое требование к надежности или скорости доставки. Например, пометка «Avia» на почтовых отправлениях в Бразилию выделит из общего потока почты в Бразилию подпоток, который будет доставляться самолетом.

---

[1] Масштабируемостью называют свойство сети допускать наращивание количества узлов и протяженность линий связи в очень широких пределах без снижения производительности.

Для каждого потока почтовая служба должна определить *маршрут*, который будет проходить через последовательность почтовых отделений, являющихся аналогами коммутаторов, то есть в данном случае решается задача маршрутизации. В результате многолетней работы почтовой службы уже определены маршруты для большинства адресов назначения. Иногда возникают новые маршруты, связанные с появлением новых возможностей — политических, транспортных, экономических. После выбора нового маршрута нужно оповестить о нем сеть почтовых отделений. Как видно, эти действия очень напоминают работу телекоммуникационной сети. Информация о выбранных маршрутах следования почты представлена в каждом почтовом отделении в виде таблицы, в которой задано соответствие между страной назначения и следующим почтовым отделением. Например, в почтовом отделении города Саратова все письма, адресованные в Индию, направляются в почтовое отделение Ашхабада, а письма, адресованные в Норвегию, — в почтовое отделение Санкт-Петербурга. Такая таблица направлений доставки почты является прямой аналогией таблицы коммутации коммуникационной сети.

Каждое почтовое отделение работает подобно *коммутатору*. Все поступающие от абонентов и других почтовых отделений почтовые отправления сортируются, то есть происходит распознавание потоков. После этого почтовые отправления, принадлежащие одному «потоку», упаковываются в мешок, для которого в соответствии с таблицей направлений определяется следующее по маршруту почтовое отделение. То есть в этом мешке *мультиплексируются* различные потоки, маршруты которых проходят через одно и то же следующее почтовое отделение. Отсортированная и упакованная корреспонденция не отправляется немедленно в следующее почтовое отделение, потому что почта для перевозки корреспонденции пользуется услугами внешнего транспортного предприятия, а оно обслуживает не только почту, но и других клиентов — магазины, склады и т. п. Поэтому почта должна ждать какое-то время, пока транспортные средства закончат перевозку других грузов и смогут забрать мешки с письмами. Транспортные средства в данном случае являются аналогом разделяемой среды компьютерной сети, а диспетчер транспортного предприятия является аналогом арбитра этой среды.

ГЛАВА 3 Коммутация каналов и пакетов

Среди множества возможных подходов к решению задачи коммутации абонентов в сетях выделяют два основополагающих, к которым относят коммутацию каналов и коммутацию пакетов.

Каждый из этих двух подходов имеет свои достоинства и недостатки. При коммутации пакетов, например, учитываются особенности компьютерного трафика, поэтому данный способ коммутации является более эффективным для компьютерных сетей по сравнению с традиционным методом коммутации каналов, применяющимся в телефонных сетях. Коммутация пакетов пытается вытеснить коммутацию каналов из традиционных для нее областей, например из телефонии (в форме интернет- или IP-телефонии), но этот спор пока не решен, и скорее всего, две техники коммутации будут сосуществовать еще долгое время, дополняя друг друга.

Коммутация каналов

Исторически коммутация каналов появилась намного раньше коммутации пакетов и ведет свое происхождение от первых телефонных сетей.

Сети, построенные по принципу коммутации каналов, имеют богатую историю, они и сегодня находят широкое применение в мире телекоммуникаций, являясь основой первичных сетей, позволяющих создавать высокоскоростные магистральные каналы связи. Первые сеансы связи между компьютерами были осуществлены через телефонную сеть, то есть также с применением техники коммутации каналов, а пользователи, которые получают доступ в Интернет по модему, продолжают обслуживаться этими сетями, так как их данные доходят до оборудования провайдера по местной телефонной сети.

Сеть с коммутацией каналов представляет собой множество **коммутаторов** и конечных узлов — **абонентов**, соединенных между собой **линиями (звеньями) связи**. Заметим, что в данном контексте термин «линия связи» используется для обозначения соединения *двух соседних узлов* сети.

В сетях с коммутацией каналов решаются все те частные задачи коммутации, которые были сформулированы ранее. Так, в качестве информационных потоков в сетях с коммутацией каналов выступают данные, которыми обмениваются пары абонентов. Время существования информационного потока ограничивается рамками **сеанса связи** абонентов. Глобальным признаком потока является пара адресов (например, телефонных номеров) абонентов, связывающихся между собой через последовательность коммутаторов.

Для всех возможных потоков заранее определяются маршруты. Маршруты в сетях с коммутацией каналов задаются либо «вручную» администратором сети, либо находятся авто-

матически с привлечением специальных программных и аппаратных средств. Маршруты фиксируются в таблицах коммутации, в которых признакам потока ставятся в соответствие идентификаторы выходных интерфейсов коммутаторов. На основании этих таблиц происходит продвижение и мультиплексирование данных. Однако, как уже было сказано, в сетях с коммутацией каналов решение всех этих задач имеет свои особенности.

Элементарный канал

Одной из особенностей сетей с коммутацией каналов является использование понятия «элементарный канал».

Элементарный канал — это базовая техническая характеристика сети с коммутацией каналов, представляющая собой некоторое фиксированное в пределах данного типа сетей значение пропускной способности. Любая линия связи в сети с коммутацией каналов имеет пропускную способность, кратную элементарному каналу, принятому для данного типа сети.

В предыдущих разделах мы использовали термин «канал» как аналог термина «линия связи». Говоря же о сетях с коммутацией каналов, мы придаем термину «канал» значение *единицы пропускной способности*.

Численное значение элементарного канала, или, другими словами, минимальная единица пропускной способности линии связи, выбирается с учетом разных факторов. Очевидно, однако, что элементарный канал не стоит выбирать меньше минимально необходимой пропускной способности для передачи ожидаемой нагрузки. Например, в современных телефонных сетях наиболее распространенным значением элементарного канала является скорость 64 Кбит/с — это минимально достаточная скорость для качественной цифровой передачи голоса.

Линии связи в сетях с коммутацией каналов (как, впрочем, и в остальных типах компьютерных сетей) имеют разную пропускную способность, одни — большую, другие — меньшую. Выбирая линии связи с разными скоростными качествами, специалисты, проектирующие сеть, стараются учесть разную интенсивность информационных потоков, которые могут возникнуть в разных фрагментах сети: чем ближе к центру сети, тем выше пропускная способность линии связи, так как магистральные линии агрегируют трафик большого количества периферийных линий связи.

Особенностью сетей с коммутацией каналов является то, что пропускная способность каждой линии связи должна быть равна *целому числу* элементарных каналов.

Так, линии связи, подключающие абонентов к телефонной сети, могут содержать 2, 24 или 30 элементарных каналов, а линии связи, соединяющие коммутаторы, — 480 или 1920 каналов.

Как следует из определения, элементарный канал представляет собой *долю пропускной способности* линии связи. В разных технологиях разделение пропускной способности выполняется по-разному. В одних случаях используется разделение по времени и элементарным каналом является пропускная способность, соответствующая одному тайм-слоту, в течение которого линия связи предоставляется некоторому потоку в исключительное

пользование. В других технологиях элементарные каналы определены как диапазоны частот, которые могут назначаться потокам.

ОЦИФРОВЫВАНИЕ ГОЛОСА

Задача оцифровывания голоса является частным случаем более общей проблемы — передачи аналоговой информации в дискретной форме. Она была решена в 60-е годы, когда голос начал передаваться по телефонным сетям в виде последовательности единиц и нулей. Такое преобразование основано на дискретизации непрерывных процессов как по амплитуде, так и по времени (рис. 3.1).

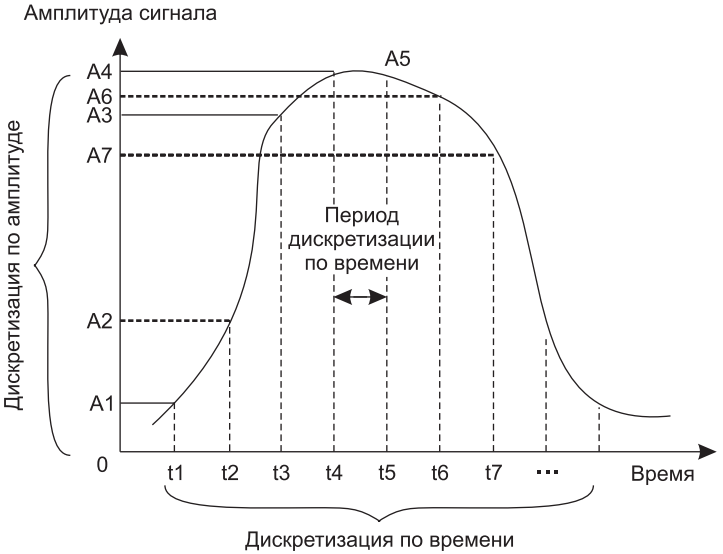


Рис. 3.1. Дискретная модуляция непрерывного процесса

Амплитуда исходной непрерывной функции измеряется с заданным периодом — за счет этого происходит *дискретизация по времени*. Затем каждый замер представляется в виде двоичного числа определенной разрядности, что означает *дискретизацию по значениям* — непрерывное множество возможных значений амплитуды заменяется дискретным множеством ее значений.

Для качественной передачи голоса используется частота квантования амплитуды звуковых колебаний в 8000 Гц (дискретизация по времени с интервалом 125 мкс). Для представления амплитуды одного замера чаще всего используется 8 бит кода, что дает 256 градаций звукового сигнала (дискретизация по значениям). В этом случае для передачи одного голосового канала необходима пропускная способность 64 Кбит/с: 8000 × 8 = 64 000 бит/с или **64 Кбит/с**. Такой голосовой канал называют **элементарным каналом цифровых телефонных сетей**.

Обратимся к фрагменту сети, изображенному на рис. 3.2. Предположим, что эта сеть характеризуется элементарным каналом *T* бит/с. В сети существуют линии связи разной пропускной способности, состоящие из 2, 3, 4 и 5 элементарных каналов. Для определенности предположим, что элементарные каналы являются дуплексными. Элементарные каналы идентифицируются номерами. Например, коммутатор S2 имеет в своем распоряжении на интерфейсе (порту) P1 элементарные каналы 1, 2, 3 и 4, а на интерфейсе P3 — элементарные каналы 1, 2, 3, 4 и 5.

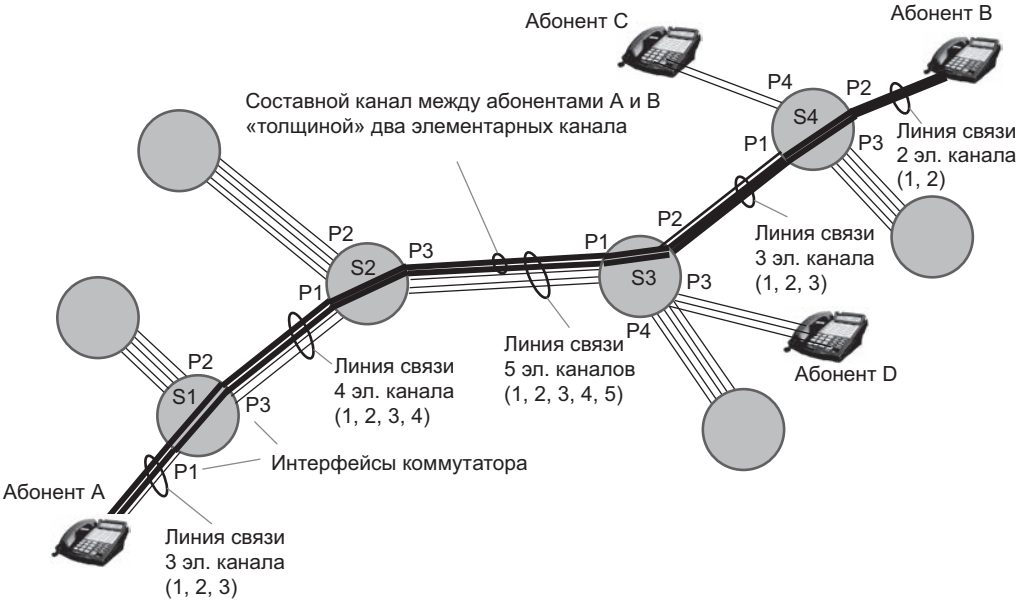


Рис. 3.2. Составной канал в сети с коммутацией каналов

На рисунке показаны два абонента, *A* и *B*, генерирующие во время сеанса связи (телефонного разговора) *информационный поток*, для которого в сети был предусмотрен *маршрут*, проходящий через четыре коммутатора — *S*1, *S*2, *S*3 и *S*4. Предположим также, что интенсивность информационного потока между абонентами не превосходит $2T$ бит/с. Тогда для обмена данными этим двум абонентам достаточно иметь в своем распоряжении по паре элементарных каналов, «выделенных» из каждой линии связи, лежащей на маршруте следования данных от пункта *A* к пункту *B*. На рисунке элементарные каналы, необходимые абонентам *A* и *B*, обозначены толстыми линиями.

Составной канал

Канал, построенный путем коммутации (соединения) выделенных для информационного потока элементарных каналов, называют **составным каналом**.

В рассматриваемом примере для соединения абонентов *A* и *B* был создан составной канал «толщиной» в два элементарных канала. Если изменить наше предположение и считать, что предложенная нагрузка гарантированно не превысит T бит/с, то абонентам будет достаточно иметь в своем распоряжении составной канал «толщиной» в один элементарный канал. В то же время абоненты, интенсивно обменивающиеся данными, могут предъявить и более высокие требования к пропускной способности составного канала. Для этого они должны в каждой линии связи зарезервировать за собой большее (но непременно одинаковое для всех линий связи) количество элементарных каналов.

Подчеркнем следующие свойства составного канала:

- составной канал на всем своем протяжении состоит из *одинакового* количества элементарных каналов;
- составной канал имеет *постоянную и фиксированную пропускную способность* на всем своем протяжении;
- составной канал создается *временно* на период сеанса связи двух абонентов или, другими словами, только на время существования потока;
- на время сеанса связи все элементарные каналы, входящие в составной канал, поступают в *исключительное* пользование абонентов, для которых был создан этот составной канал;
- в течение всего сеанса связи абоненты могут посылать в сеть данные со скоростью, не превышающей пропускную способность составного канала;
- данные, поступившие в составной канал, гарантированно доставляются вызываемому абоненту *без задержек, потерь и с той же скоростью* (скоростью источника) вне зависимости от того, существуют ли в это время в сети другие соединения или нет;
- после окончания сеанса связи элементарные каналы, входившие в соответствующий составной канал, *объявляются свободными* и возвращаются в пул распределяемых ресурсов для использования другими абонентами.

В сети может одновременно происходить несколько сеансов связи (обычная ситуация для телефонной сети, в которой одновременно передаются разговоры сотен и тысяч абонентов). Разделение сети между сеансами связи происходит на уровне элементарных каналов. Например (см. рис. 3.2), мы можем предположить, что после того, как в линии связи *S*2–*S*3 было выделено два канала для связи абонентов *A* и *B*, оставшиеся три элементарных канала были распределены между тремя другими сеансами связи, проходившими в это же время и через эту же линию связи. Такое *мультиплексирование* позволяет одновременно передавать через каждый физический канал трафик нескольких логических соединений.

Мультиплексирование означает, что абоненты вынуждены конкурировать за ресурсы, в данном случае за элементарные каналы. Возможны ситуации, когда некоторая промежуточная линия связи уже исчерпала свободные элементарные каналы — тогда новый сеанс связи, маршрут которого пролегает через данную линию связи, не может состояться.

Чтобы распознать такие ситуации, обмен данными в сети с коммутацией каналов предваряется **процедурой установления соединения**. В соответствии с этой процедурой абонент, являющийся инициатором сеанса связи (например, абонент *A* в нашей сети), посылает в коммутационную сеть **запрос**, представляющий собой сообщение, содержащее адрес вызываемого абонента, например абонента *B*[1].

Цель запроса — проверить, можно ли образовать составной канал между вызывающим и вызываемым абонентами. Для этого требуется соблюдение двух условий: наличие требуемого числа свободных элементарных каналов в каждой линии связи, лежащей на пути от *A* к *B*, и незанятость вызываемого абонента в другом соединении.

Запрос перемещается по *маршруту*, заранее определенному для информационного потока данной пары абонентов. Этот маршрут зафиксирован в *глобальных таблицах коммутации*, размещенных на всех промежуточных коммутаторах на пути от абонента *A* к *B*.

---

[1] В телефонной сети посылке запроса соответствует набор телефонного номера.

Например, коммутатор *S*4 в нашем примере может иметь следующую глобальную таблицу маршрутизации:

Таблица 3.1. Пример глобальной таблицы маршрутизации коммутатора S4

Абонент	Интерфейс, ведущий к следующему коммутатору
A	P1
B	P2
C	P4
D	P1

Если в результате прохождения запроса выяснилось, что ничто не препятствует установлению соединения, то происходит *фиксация составного канала*. Для этого в каждом из коммутаторов вдоль пути от *A* до *B* создаются записи в *локальных таблицах коммутации*, в которых указывается соответствие между *локальными признаками потока* — номерами элементарных каналов, зарезервированных для этого сеанса связи в данном коммутаторе.

В нашем примере после прохождения запроса на установление связи от абонента *A* к абоненту *B* в каждом из коммутаторов *S*1, *S*2, *S*3 и *S*4 были созданы соответствующие записи в их локальных таблицах коммутации. В коммутаторе *S*4 локальная таблица выглядит так:

Таблица 3.2. Пример локальной таблицы маршрутизации

Входные элементарные каналы	Выходные элементарные каналы
P1/канал 2	P2/канал 1
P1/канал 3	P2/канал 2

Представленные в примере записи говорят о том, что элементарный канал 2, поступающий на интерфейс *P*1, скоммутирован с элементарным каналом 1 интерфейса *P*2, а элементарный канал 3 интерфейса *P*1 — с элементарным каналом 2 интерфейса *P*2.

В этом примере номера элементарных каналов имеют уникальные значения в пределах каждого интерфейса, поэтому в локальной таблице коммутации элементарные каналы идентифицируются парой номер интерфейса/номер канала. Возможна и другая организация коммутатора, когда номера элементарных каналов имеют уникальные значения в пределах коммутатора, но при добавлении или удалении интерфейса их приходится перенумеровывать, что не всегда удобно.

После того как составной канал установлен и в каждом коммутаторе сформированы соответствующие записи в локальных таблицах коммутации, абоненты *A* и *B* могут начать свой сеанс связи.

Таким образом, продвижение данных в сетях с коммутацией каналов происходит в два этапа:

1. В сеть поступает служебное сообщение — запрос, который несет адрес вызываемого абонента и инициирует создание составного канала.
2. По подготовленному составному каналу передается основной поток данных, для передачи которого уже не требуется никакой вспомогательной информации, в том числе адреса вызываемого абонента. Коммутация данных в коммутаторах выполняется на основе локальных признаков потока — номеров выделенных ему элементарных каналов.

Запросы на установление соединения не всегда завершаются успешно. Если на пути между вызывающим и вызываемым абонентами отсутствуют свободные элементарные каналы или вызываемый узел занят, то происходит **отказ в установлении соединения**. Например, если во время сеанса связи абонентов *A* и *B* абонент *C* пошлет запрос в сеть на установление соединения с абонентом *D*, то он получит отказ, потому что из двух необходимых ему элементарных каналов линии связи *S3*–*S4* свободным является только один (рис. 3.3). При отказе в установлении соединения сеть информирует вызывающего абонента специальным сообщением. Чем больше нагрузка на сеть, то есть чем больше соединений она в данный момент поддерживает, тем больше вероятность отказа в удовлетворении запроса на установление нового соединения.

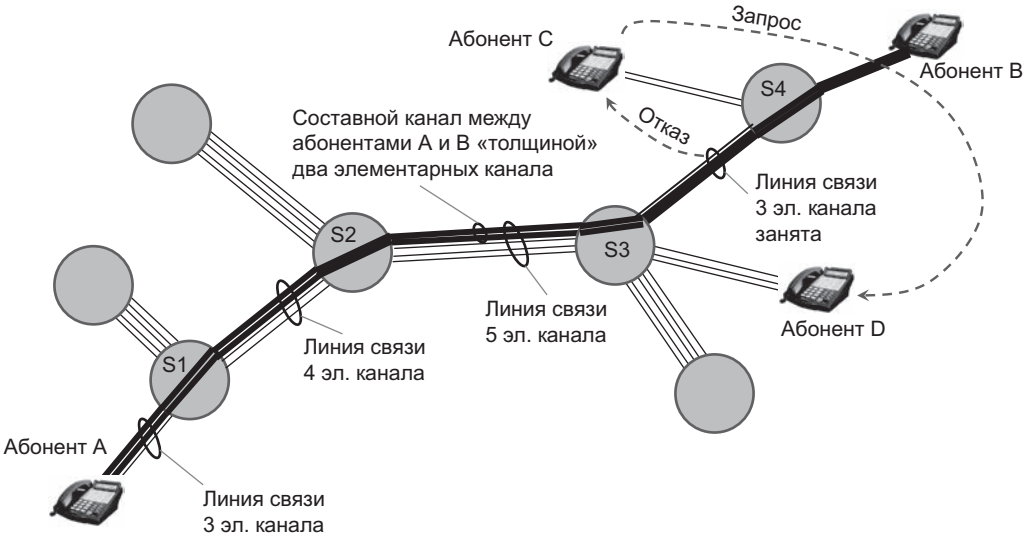


Рис. 3.3. Отказ в установлении соединения в сети с коммутацией каналов

Мы описали процедуру установления соединения в *динамическом режиме*, когда соединение устанавливается немедленно по запросу абонента сети и только на время сеанса связи. Подобный режим используется телефонными сетями: телефонный аппарат генерирует запрос, посылая в сеть импульсы (или тоновые сигналы), кодирующие номер вызываемого абонента, а сеть либо устанавливаст соединение, либо сообщает об отказе сигналами «занято».

Однако это не единственно возможный режим работы сети с коммутацией каналов. Существует и *статический режим* установления соединения, характерный для случаев, когда необходимо установить составной канал не на время одного сеанса связи абонентов, а на более долгий срок. Такие долговременные каналы создаются не по инициативе абонентов, а заранее администратором сети. Очевидно, что статический режим малопригоден для традиционной телефонной сети с ее короткими сеансами связи, однако он вполне оправдан для создания высокоскоростных телекоммуникационных каналов между городами и странами на более-менее постоянной основе.

Технология коммутации каналов ориентирована на минимизацию случайных событий в сети, то есть это технология, стремящаяся к детерминизму. Во избежание всевозможных

неопределенностей значительная часть работы по организации информационного обмена выполняется еще до того, как начнется собственно передача данных. Сначала по заданному адресу проверяется доступность необходимых элементарных каналов на всем пути от отправителя до адресата. Затем эти каналы закрепляются на все время сеанса для исключительного использования двумя абонентами и коммутируются в один непрерывный «трубопровод» (составной канал), имеющий «шлюзовые задвижки» на стороне каждого из абонентов. После этой исчерпывающей подготовительной работы остается сделать самое малое: «открыть шлюзы» и позволить информационному потоку свободно и без помех «перетекать» между заданными точками сети (рис. 3.4).

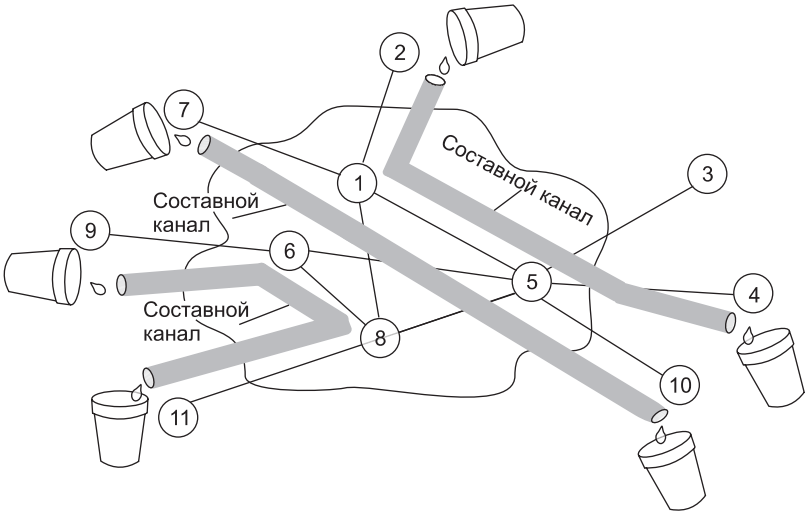


Рис. 3.4. Сеть с коммутацией каналов как система трубопроводов

Неэффективность передачи пульсирующего трафика

Сети с коммутацией каналов наиболее эффективно передают пользовательский трафик в том случае, когда скорость его постоянна в течение всего сеанса связи и максимально соответствует *фиксированной* пропускной способности физических линий связи сети. Эффективность работы сети снижается, когда информационные потоки, генерируемые абонентами, приобретают *пульсирующий* характер.

Так, разговаривая по телефону, люди постоянно меняют темп речи, перемежая быстрые высказывания паузами. В результате соответствующие «голосовые» информационные потоки становятся неравномерными, а значит, снижается эффективность передачи данных. Правда, в случае телефонных разговоров это снижение оказывается вполне приемлемым и позволяет широко использовать сети с коммутацией каналов для передачи голосового трафика.

Гораздо сильнее снижает эффективность сети с коммутацией каналов передача так называемого *компьютерного трафика*, то есть трафика, генерируемого приложениями, с кото-

рыми работает пользователь компьютера. Этот трафик практически всегда является пульсирующим. Например, когда вы загружаете из интернета очередную страницу, скорость трафика резко возрастает, а после окончания загрузки падает практически до нуля. Если для описанного сеанса доступа в интернет вы задействуете сеть с коммутацией каналов, то большую часть времени составной канал между вашим компьютером и веб-сервером будет простаивать. В то же время часть пропускной способности сети окажется закрепленной за вами и останется недоступной другим пользователям сети. Сеть в такие периоды похожа на пустой эскалатор метро, который движется, но полезную работу не выполняет.

Для эффективной передачи неравномерного компьютерного трафика была специально разработана техника коммутации пакетов.

Коммутация пакетов

Сети с коммутацией пакетов, так же как и сети с коммутацией каналов, состоят из коммутаторов, связанных физическими линиями связи. Однако передача данных в этих сетях происходит совершенно по-другому. Образно говоря, по сравнению с сетью с коммутацией каналов сеть с коммутацией пакетов ведет себя менее «ответственно». Например, она может принять данные для передачи, не заботясь о резервировании линий связи на пути следования этих данных и не гарантируя требуемую пропускную способность. Сеть с коммутацией пакетов не создает заранее для своих абонентов отдельных каналов связи, выделенных исключительно для них. Данные могут задерживаться и даже теряться по пути следования. Как же при таком хаосе и неопределенности сеть с коммутацией пакетов выполняет свои функции по передаче данных?

Важнейшим принципом функционирования сетей с коммутацией пакетов является представление информации, передаваемой по сети , в виде структурно отделенных друг от друга порций данных, называемых **пакетами**[1].

Каждый пакет снабжен **заголовком** (рис. 3.5), в котором содержится адрес назначения и другая вспомогательная информация (длина поля данных, контрольная сумма и др.), используемая для доставки пакета адресату. Наличие адреса в каждом пакете является одной из важнейших особенностей техники коммутации пакетов, так как каждый пакет может быть обработан коммутатором *независимо* от других пакетов, составляющих сетевой трафик. Однако независимость обработки пакетов не является неотъемлемым свойством технологии коммутации пакетов. В некоторых технологиях коммутации пакетов от полной независимости пакетов отказываются для достижения определенного преимущества — например, повышения надежности доставки пакетов или же повышения эффективности их прохождения через сеть.

Помимо заголовка у пакета может иметься еще одно дополнительное поле, размещаемое в конце пакета и поэтому называемое **концевиком**. В концевике обычно помещается **контрольная сумма**, которая позволяет проверить, была ли искажена информация при передаче через сеть или нет.

---

[1] Наряду с термином «пакет» используются также термины «кадр», «фрейм», «ячейка» и др. В данном контексте различия в значении этих терминов несущественны.

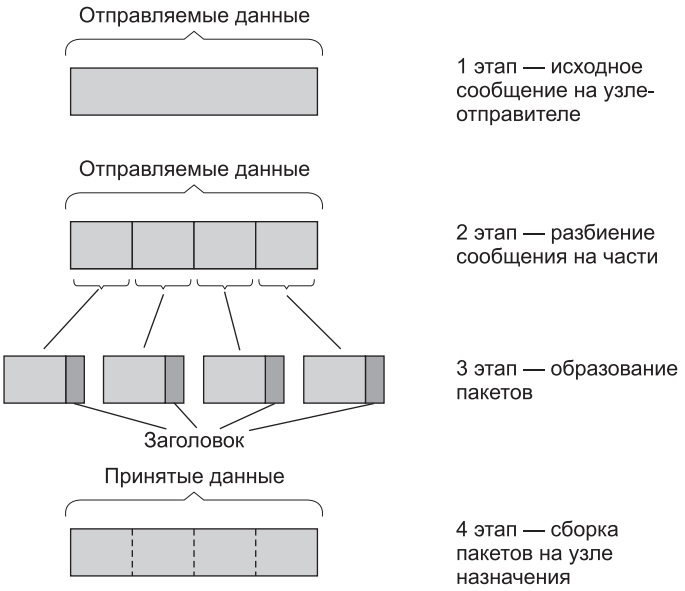


Рис. 3.5. Разбиение данных на пакеты

В зависимости от конкретной реализации технологии коммутации пакетов последние могут иметь фиксированную или переменную длину. Кроме того, может меняться состав информации, размещенной в заголовках пакетов. Например, в технологии ATM пакеты (называемые там ячейками) имеют фиксированную длину, а в технологии Ethernet установлены лишь минимально и максимально возможные размеры пакетов (кадров).

Пакеты поступают в сеть в том темпе, в котором их генерирует источник. Предполагается, что сеть с коммутацией пакетов, в отличие от сети с коммутацией каналов, всегда готова принять пакет от конечного узла.

Как и в сетях с коммутацией каналов, в сетях с коммутацией пакетов для каждого из потоков вручную или автоматически определяется маршрут, фиксируемый в хранящихся на коммутаторах таблицах коммутации. Пакеты, попадая на коммутатор, обрабатываются и направляются по тому или иному маршруту на основании информации, содержащейся в их заголовках, а также в таблице коммутации (рис. 3.6).

Пакеты, принадлежащие как одному и тому же, так и разным информационным потокам, при перемещении по сети могут «перемешиваться» между собой, образовывать очереди и «тормозить» друг друга. На пути пакетов могут встречаться линии связи, имеющие разную пропускную способность. В зависимости от времени суток может сильно меняться и степень загруженности линий связи. В таких условиях не исключены ситуации, когда пакеты, принадлежащие одному и тому же потоку, могут перемещаться по сети с разными скоростями и даже прийти к месту назначения не в том порядке, в котором они были отправлены.

Разделение данных на пакеты позволяет передавать неравномерный компьютерный трафик более эффективно, чем в сетях с коммутацией каналов. Это объясняется тем, что пульсации трафика от отдельных компьютеров носят случайный характер и распределяются во времени так, что их пики чаще всего не совпадают. Поэтому когда линия связи передает

трафик большого количества конечных узлов, в суммарном потоке пульсации сглаживаются и пропускная способность линии используется более рационально, без длительных простоев. Так, на рис. 3.7 показаны неравномерные потоки пакетов, поступающие от конечных узлов 3, 4 и 10 в сети, изображенной на рис. 3.6. Предположим, что эти потоки передаются в направлении коммутатора 8, а следовательно, накладываются друг на друга при прохождении линии связи между коммутаторами 5 и 8. Получающийся в результате суммарный поток является более равномерным, чем каждый из образующих его отдельных потоков.

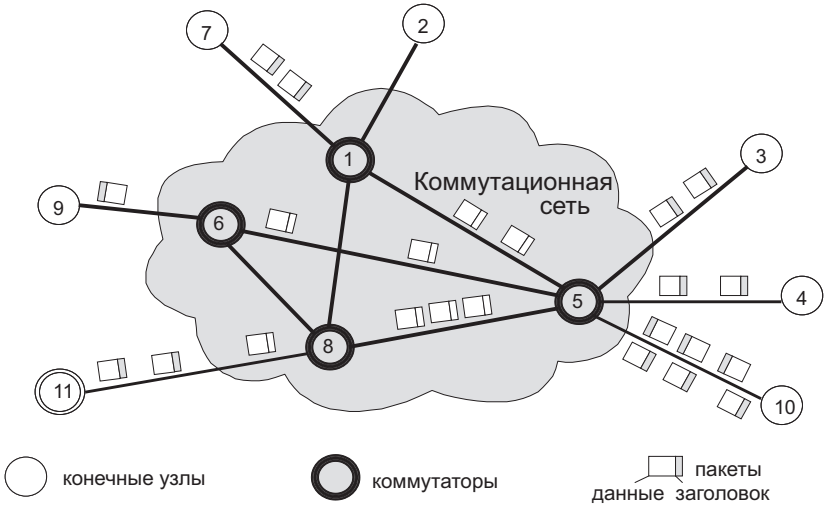


Рис. 3.6. Передача данных по сети в виде пакетов

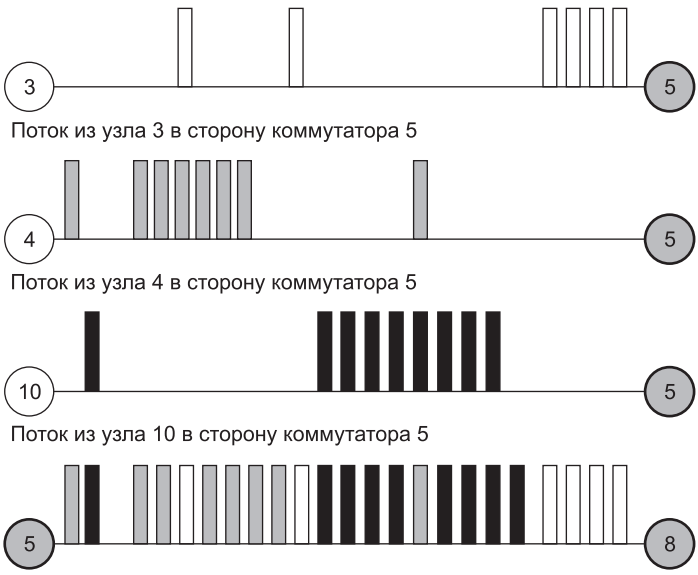


Рис. 3.7. Сглаживание трафика в сетях с коммутацией пакетов

Буферизация пакетов

Неопределенность и асинхронность перемещения данных в сетях с коммутацией пакетов предъявляет особые требования к работе коммутаторов в таких сетях.

> Главное отличие пакетных коммутаторов[1] от коммутаторов в сетях с коммутацией каналов состоит в том , что они имеют внутреннюю **буферную память** для временного хранения пакетов.

Действительно, пакетный коммутатор не может принять решения о продвижении пакета, не имея в своей памяти всего пакета. Коммутатор проверяет контрольную сумму; если она говорит о том, что данные пакета не искажены, то коммутатор начинает обрабатывать пакет и по адресу назначения определяет следующий коммутатор. Поэтому *каждый* пакет последовательно, бит за битом, помещается во **входной буфер**. Имея в виду это свойство, говорят, что сети с коммутацией пакетов используют технику **сохранения с продвижением** (store-and-forward). Заметим, что для этой цели коммутатору было бы достаточно иметь буфер размером в один пакет.

Но коммутатору нужны буферы и для других целей, в частности, *для согласования скоростей передачи данных в линиях связи*, подключенных к его интерфейсам. Действительно, если скорость поступления потока пакетов из одной линии связи в течение некоторого периода превышает пропускную способность той линии связи, в которую эти пакеты должны быть направлены, то во избежание потерь пакетов на целевом интерфейсе необходимо организовать выходную очередь (рис. 3.8).

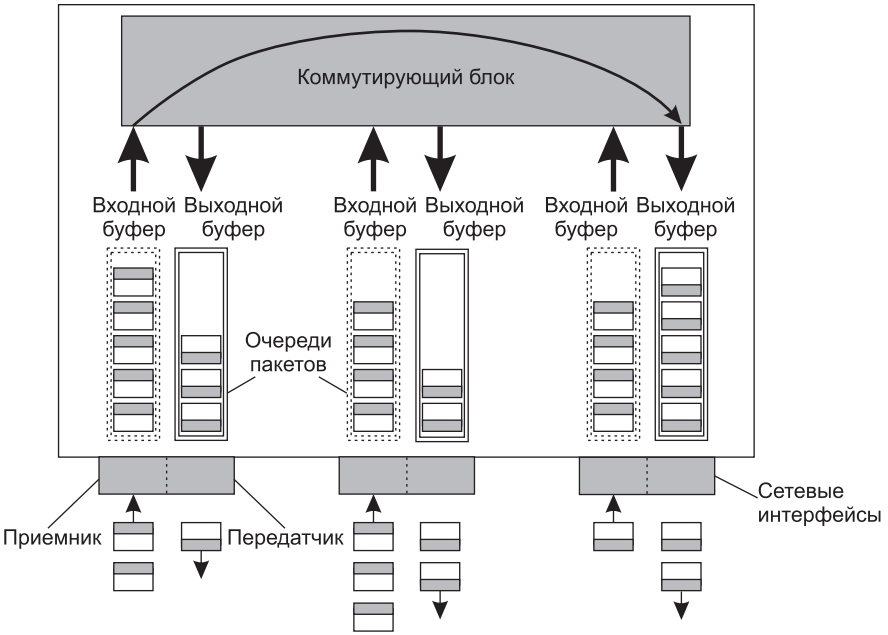


Рис. 3.8. Буферы и очереди пакетов в коммутаторе

[1] Иногда мы будем называть коммутаторы сетей с коммутацией пакетов пакетными коммутаторами, а сети с коммутацией пакетов — пакетными сетями.

Буферизация необходима пакетному коммутатору и *для согласования скорости поступления пакетов со скоростью их коммутации*. Если коммутирующий блок не успевает обрабатывать пакеты (анализировать заголовки и перебрасывать пакеты на нужный интерфейс), то на интерфейсах коммутатора возникают **входные очереди**. Очевидно, что для хранения входной очереди объем буфера должен превышать размер одного пакета.

Существуют различные подходы к построению коммутирующего блока. Традиционный способ основан на одном центральном процессоре, который обслуживает все входные очереди коммутатора, что может приводить к большим очередям, так как производительность процессора разделяется между несколькими очередями. Современные способы построения коммутирующего блока основаны на многопроцессорном подходе, когда каждый интерфейс имеет свой встроенный процессор для обработки пакетов. Кроме того, существует центральный процессор, координирующий работу интерфейсных процессоров, использование которых повышает производительность коммутатора и уменьшает очереди на входных интерфейсах. Однако такие очереди все равно могут возникать, так как центральный процессор по-прежнему остается узким местом.

Поскольку объем буферов в коммутаторах ограничен, иногда происходит *потеря пакетов* из-за переполнения буферов при временной перегрузке части сети, когда, например, совпадают периоды пульсации нескольких информационных потоков. Для компенсации таких потерь в технологии коммутации пакетов предусмотрен ряд специальных механизмов.

Более подробно работа коммутатора обсуждается в главах 10 и 11.

Пакетный коммутатор может работать на основании одного из трех методов продвижения пакетов:

- дейтаграммная передача;
- передача с установлением логического соединения;
- передача с установлением виртуального канала.

Дейтаграммная передача

Дейтаграммный способ передачи данных основан на том, что все передаваемые пакеты *продвигаются* (передаются от одного узла сети другому) *независимо* друг от друга на основании одних и тех же правил. Никакая информация об *уже переданных пакетах* сетью не хранится и в ходе обработки очередного пакета во внимание не принимается. То есть каждый отдельный пакет рассматривается сетью как совершенно независимая единица передачи — **дейтаграмма**.

Решение о продвижении пакета принимается на основе таблицы коммутации, ставящей в соответствие адресам назначения пакетов информацию, однозначно определяющую следующий по маршруту транзитный (или конечный) узел. В качестве такой информации могут выступать идентификаторы интерфейсов данного коммутатора или адреса входных интерфейсов коммутаторов, следующих по маршруту.

На рис. 3.9 показана сеть, в которой шесть конечных узлов ($N1$–$N6$) связаны семью коммутаторами ($S1$–$S7$). Показаны также несколько перемещающихся по разным маршрутам пакетов с разными адресами назначения ($N1$–$N6$), на пути которых лежит коммутатор $S1$. При поступлении каждого из этих пакетов в коммутатор $S1$ выполняется просмотр соответствующей таблицы коммутации и выбор дальнейшего пути перемещения. Так, пакет

с адресом *N*5 будет передан коммутатором *S*1 на интерфейс, ведущий к коммутатору *S*6, где в результате подобной процедуры этот пакет будет направлен конечному узлу получателя *N*5.

Таблица коммутации коммутатора S1

Адрес назначения	Адрес следующего коммутатора
N1	Пакет не требуется передавать через сеть
N2	S2
N2	S3
N3	S3
N4	S3
N5	S6
N6	S6

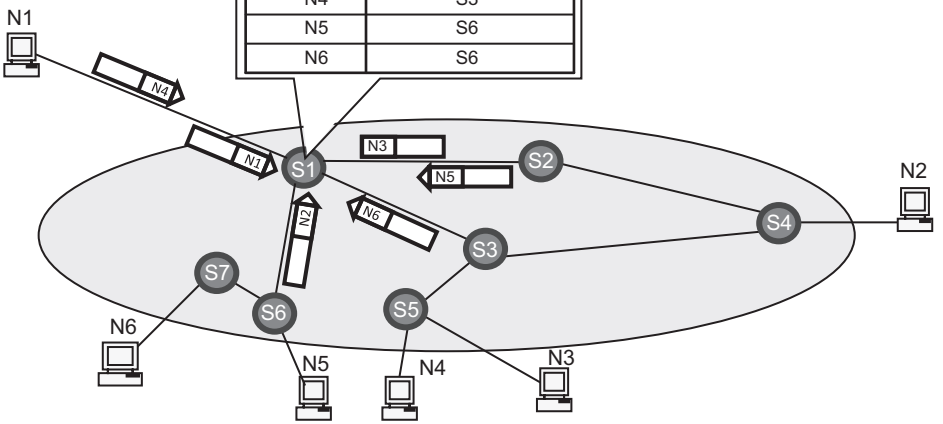


Рис. 3.9. Иллюстрация дейтаграммного принципа передачи пакетов

В таблице коммутации для одного и того же адреса назначения может содержаться несколько записей, указывающих соответственно на различные адреса следующего коммутатора. Такой подход называется **балансом нагрузки** и используется для повышения производительности и надежности сети. Как видим, на рис. 3.9 пакеты, поступающие в коммутатор *S*1 для узла назначения с адресом *N*2, распределяются между двумя следующими коммутаторами — *S*2 и *S*3, что снижает нагрузку на каждый из них, а значит, сокращает очереди и ускоряет доставку. Некоторая «размытость» путей следования пакетов с одним и тем же адресом назначения через сеть является прямым следствием принципа независимой обработки каждого пакета, присущего дейтаграммному методу. Пакеты, следующие по одному и тому же адресу назначения, могут добираться до него разными путями также вследствие изменения состояния сети, например отказа промежуточных коммутаторов.

Дейтаграммный метод работает быстро, так как никаких предварительных действий перед отправкой данных проводить не требуется. Однако при таком методе трудно проверить факт доставки пакета узлу назначения.

> В дейтаграммном методе доставка пакета не гарантируется, а выполняется по мере возможности — для описания такого свойства используется термин **доставка по возможности (best effort)**.

Передача с установлением логического соединения

Следующий рассматриваемый нами способ продвижения пакетов основывается на знании устройствами сети «истории» обмена данными, например, на запоминании узлом-отправителем числа отправленных, а узлом-получателем — числа полученных пакетов. Такого рода информация фиксируется в рамках логического соединения.

> Процедура согласования двумя конечными узлами сети некоторых параметров процесса обмена пакетами называется **установлением логического соединения**. Параметры, о которых договариваются два взаимодействующих узла, называются **параметрами логического соединения**.

Наличие логического соединения позволяет более рационально по сравнению с дейтаграммным способом обрабатывать пакеты. Например, при потере нескольких предыдущих пакетов может быть снижена скорость отправки последующих. Напротив, благодаря нумерации пакетов и отслеживанию номеров отправленных и принятых пакетов можно повысить надежность путем отбрасывания дубликатов, упорядочивания поступивших и повторения передачи потерянных пакетов.

Параметры соединения могут быть *постоянными*, то есть не изменяющимися в течение всего соединения (например, идентификатор соединения, способ шифрования пакета или максимальный размер поля данных пакета), или *переменными*, то есть динамически отражающими текущее состояние соединения (например, последовательные номера передаваемых пакетов).

Когда отправитель и получатель *фиксируют* начало нового соединения, они прежде всего «договариваются» о начальных значениях параметров процедуры обмена и только после этого начинают передачу собственно данных.

Передача с установлением логического соединения включает три фазы (рис. 3.10):

- *Установление логического соединения*, которое начинается с того, что узел — инициатор соединения отправляет узлу-получателю служебный пакет с предложением установить соединение. Если узел-получатель согласен с этим, то он посылает в ответ другой служебный пакет, подтверждающий установление соединения и предлагающий некоторые параметры, которые должны использоваться в рамках данного логического соединения. Это могут быть, например, идентификатор соединения, количество пакетов, которые можно отправить без получения подтверждения, параметры метода шифрования и т. п.

  Узел — инициатор соединения может закончить процесс установления соединения отправкой третьего служебного пакета, в котором сообщит, что предложенные параметры ему подходят.

- *Передача данных*. После того как соединение установлено и все параметры согласованы, конечные узлы начинают передачу собственно данных. Логическое соединение может быть рассчитано на передачу данных как в одном направлении (от инициатора соединения), так и в обоих направлениях.

- *Разрыв логического соединения.* После передачи некоторого законченного набора данных, например определенного файла, узел-отправитель инициирует процедуру разрыва логического соединения, аналогичную процедуре установления соединения.

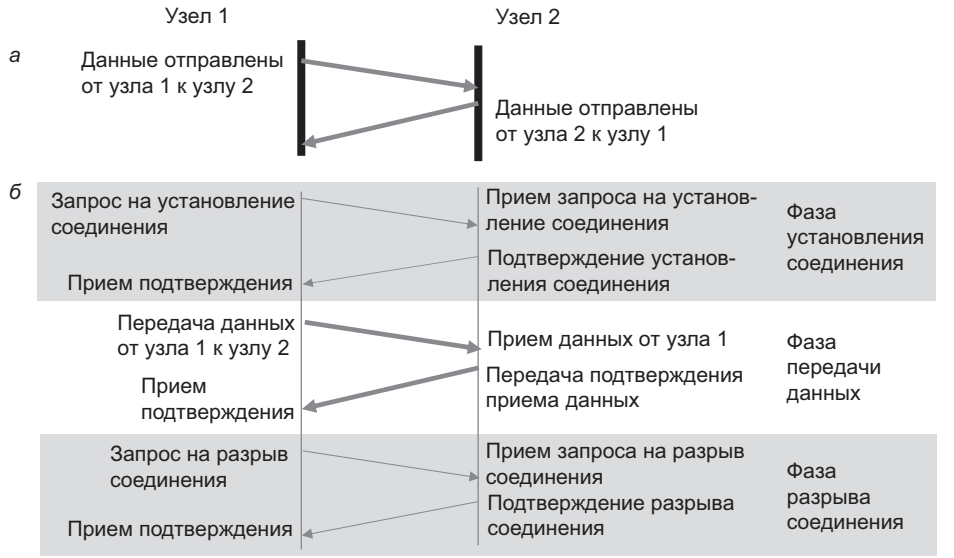


Рис. 3.10. Передача без установления соединения (*а*) и с установлением соединения (*б*)

Заметим, что, в отличие от передачи дейтаграммного типа, в которой поддерживается только один тип пакетов — информационный, передача с установлением соединения должна поддерживать как минимум два типа пакетов — *информационные* пакеты переносят собственно пользовательские данные, а *служебные* предназначаются для установления (разрыва) соединения.

Информационные пакеты обрабатываются коммутаторами точно так же, как и при дейтаграммной передаче: из заголовков пакетов извлекаются адреса назначения и сравниваются с записями в таблицах коммутации, содержащими информацию о следующих шагах по маршруту. Как и дейтаграммы, пакеты, относящиеся к одному логическому соединению, в некоторых случаях (например, при отказе линии связи) могут доставляться адресату по разным маршрутам.

Однако передача с установлением соединения имеет важное отличие от дейтаграммной передачи, поскольку в ней, помимо обработки пакетов на коммутаторах, имеет место *дополнительная обработка пакетов на конечных узлах*. Например, если при установлении соединения была оговорена передача данных в зашифрованном виде, то шифрование пакетов выполняется узлом-отправителем, а дешифрование — узлом-получателем. Аналогично, для обеспечения в рамках логического соединения надежности всю работу по нумерации пакетов, отслеживанию номеров доставленных и недоставленных пакетов, отправке копий и отбрасыванию дубликатов берут на себя конечные узлы.

Механизм установления логических соединений позволяет реализовывать дифференцированное обслуживание информационных потоков. Разное обслуживание могут получить даже потоки, относящиеся к одной и той же паре конечных узлов. Например, пара конечных узлов может установить два параллельно работающих логических соединения, в одном из которых передавать данные в зашифрованном виде, а в другом — открытым текстом.

Как видим, передача с установлением соединения предоставляет больше возможностей в плане надежности и безопасности обмена данными, чем дейтаграммная передача. Однако этот способ более медленный, так как он подразумевает дополнительные вычислительные затраты на установление и поддержание логического соединения.

Передача с установлением виртуального канала

Следующий способ продвижения данных в пакетных сетях основан на частном случае логического соединения, в число параметров которого входит жестко определенный для всех пакетов *маршрут*. То есть все пакеты, передаваемые в рамках данного соединения, должны проходить строго по одному и тому же закрепленному за этим соединением пути.

Единственный заранее проложенный фиксированный маршрут, соединяющий конечные узлы в сети с коммутацией пакетов, называют **виртуальным каналом** (virtual circuit, или virtual channel).

Виртуальные каналы прокладываются для *устойчивых* информационных потоков. С целью выделения потока данных из общего трафика каждый пакет этого потока помечается признаком особого вида — **меткой**.

Как и в сетях с установлением логических соединений, прокладка виртуального канала начинается с отправки узлом-источником специального пакета — запроса на установление соединения. В запросе указываются адрес назначения и метка потока, для которого прокладывается этот виртуальный канал. Запрос, проходя по сети, формирует новую запись в таблице каждого из коммутаторов, расположенных на пути от отправителя до получателя. Запись говорит о том, каким образом коммутатор должен обслуживать пакет, имеющий заданную метку. Образованный виртуальный канал идентифицируется той же меткой[1].

После прокладки виртуального канала сеть может передавать по нему соответствующий поток данных. Во всех пакетах, которые переносят пользовательские данные, адрес назначения уже не указывается, вместо него используется метка виртуального канала. При поступлении пакета на входной интерфейс коммутатор читает значение метки из заголовка пришедшего пакета и просматривает свою таблицу коммутации, по которой определяет, на какой выходной порт передать пришедший пакет.

На рис. 3.11 показана сеть, в которой проложено два виртуальных канала, идентифицируемых метками VC1 и VC2. Первый проходит от конечного узла с адресом *N*1 до конечного узла с адресом *N*2 через промежуточные коммутаторы *S*1, *S*2 и *S*4. Второй виртуальный канал VC2 обеспечивает продвижение данных по пути *N*1–*S*1–*S*3–*S*5–*N*3. В общем случае между двумя конечными узлами может быть проложено несколько виртуальных каналов, например, еще один виртуальный канал между узлами *N*1 и *N*2 мог бы проходить через промежуточный коммутатор *S*3. На рисунке показаны два пакета, несущие в своих заголовках метки потоков VC1 и VC2, которые играют роли адресов назначения.

---

[1] Эта метка в различных технологиях называется по-разному, например, номером логического канала (Logical Channel Number, LCN) в технологии X.25 или просто «меткой» в технологии MPLS.

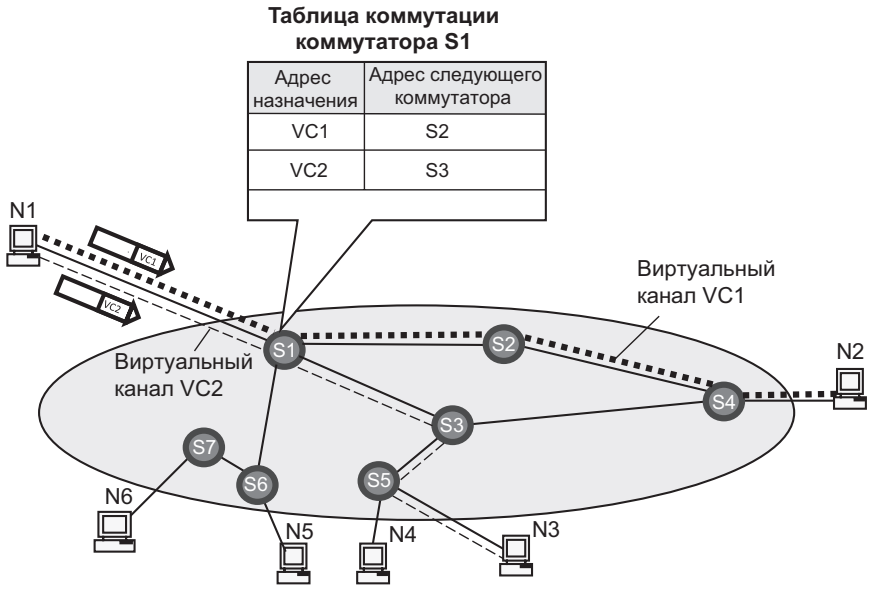


Рис. 3.11. Иллюстрация принципа работы виртуального канала

Таблица коммутации в сетях, использующих виртуальные каналы, отличается от таблицы коммутации в дейтаграммных сетях. Она содержит записи *только о проходящих через коммутатор виртуальных каналах*, а не обо всех возможных адресах назначения, как это имеет место в сетях с дейтаграммным алгоритмом продвижения. Обычно в крупной сети количество проложенных через узел виртуальных каналов существенно меньше общего количества узлов, поэтому таблицы коммутации в этом случае намного короче и, следовательно, анализ такой таблицы занимает у коммутатора меньше времени. По той же причине метка короче адреса конечного узла, и заголовок пакета в сетях с виртуальными каналами переносит по сети вместо длинного адреса компактный идентификатор потока.

ВНИМАНИЕ

Использование в сетях техники виртуальных каналов или логических соединений не делает их сетями с коммутацией каналов. Хотя в подобных сетях также применяется процедура предвари тельного установления канала, этот канал является виртуальным, то есть по нему передаются отдельные пакеты, а не потоки информации с постоянной скоростью, как в сетях с коммутацией каналов.

В одной и той же сетевой технологии могут быть задействованы разные способы продвижения данных. Так, для передачи данных между отдельными сетями, составляющими интернет, используется дейтаграммный протокол IP. В то же время обеспечением надежной доставки данных между конечными узлами этой сети занимается протокол TCP, устанавливающий логические соединения без фиксации маршрута. И наконец, интернет — это пример сети, применяющей технику виртуальных каналов, так как в состав интернета входит немало сетей MPLS, поддерживающих виртуальные каналы.

Сравнение сетей с коммутацией пакетов и каналов

Сравним основные свойства сетей с коммутацией каналов и дейтаграммных сетей с коммутацией пакетов.

Таблица 3.3. Сравнение сетей с коммутацией каналов и дейтаграммных сетей с коммутацией пакетов

Коммутация каналов	Коммутация пакетов
Необходимо предварительно устанавливать соединение	Отсутствует этап установления соединения
Адрес требуется только на этапе установления соединения	Адрес и другая служебная информация передаются с каждым пакетом
Сеть может отказать абоненту в установлении соединения	Сеть всегда готова принять данные от абонента
Гарантированная пропускная способность (полоса пропускания) для взаимодействующих абонентов	Пропускная способность сети для абонентов неизвестна, задержки передачи носят случайный характер
Достоинством сети является то, что трафик реального времени передается без задержек	Достоинством сети является то, что ресурсы сети используются эффективно при передаче пульсирующего трафика
Надежность передачи: высокая	Надежность передачи: возможны потери данных из-за переполнения буферов
Нерациональное использование пропускной способности каналов, снижающее общую эффективность сети	Автоматическое динамическое распределение пропускной способности физического канала между абонентами

Прежде чем далее проводить техническое сравнение сетей с коммутацией пакетов и сетей с коммутацией каналов, проведем их неформальное сравнение на основе, как представляется, весьма продуктивной транспортной аналогии.

Транспортная аналогия для сетей с коммутацией пакетов и каналов

Для начала убедимся, что движение на дорогах имеет много общего с перемещением пакетов в сети *с коммутацией пакетов*.

Пусть автомобили в этой аналогии соответствуют пакетам, дороги — каналам связи, а перекрестки — коммутаторам. Подобно пакетам, автомобили перемещаются независимо друг от друга, разделяя пропускную способность дорог и создавая препятствия друг другу. Неравномерное движение автомобилей и слишком интенсивный трафик, не соответствующий пропускной способности дорог, приводит к их перегруженности, в результате автомобили стоят в пробках, что соответствует очередям пакетов в коммутаторах.

На перекрестках происходит «коммутация» потоков автомобилей, каждый из автомобилей выбирает подходящее направление перекрестка, чтобы попасть в пункт назначения. Конечно, перекресток играет намного более пассивную роль по сравнению с коммутатором пакетов. Его активное участие в обработке трафика можно заметить только на регулируемых перекрестках, где очередность пересечения перекрестка потоками автомобилей определяет светофор. Еще активнее, естественно, поведение регулировщика трафика, который может

выбрать для продвижения не только поток автомобилей в целом, но и отдельный автомобиль.

Теперь попробуем найти общее у автомобильного движения и сетей с *коммутацией каналов*.

Иногда на дороге возникает ситуация, когда нужно обеспечить особые условия для движения колонны автомобилей. Например, представим, что очень длинная колонна автобусов перевозит детей из города в летний лагерь по многополосному шоссе. Чтобы колонна двигалась без препятствий, для ее движения заранее разрабатывается маршрут.

Затем на протяжении всего этого маршрута, который пересекает несколько перекрестков, для колонны выделяется отдельная полоса на всех отрезках шоссе. При этом полоса освобождается от другого трафика уже за некоторое время до начала движения колонны, а отменяется это резервирование только после того, как колонна достигает пункта назначения.

Во время движения все автомобили колонны едут с одинаковой скоростью и приблизительно равными интервалами между собой, не создавая препятствий друг другу. Очевидно, что для колонны автомобилей создаются наиболее благоприятные условия движения, но при этом автомобили теряют свою самостоятельность, превращаясь в поток, из которого нельзя свернуть в сторону. Дорога при такой организации движения используется нерационально — полоса простаивает значительную часть времени, как и полоса пропускания в сетях с коммутацией каналов.

Структура задержек в сетях с коммутацией каналов и пакетов

Вернемся от автомобилей к сетевому трафику. Пусть пользователю сети необходимо передать достаточно неравномерный трафик, состоящий из периодов активности и пауз. Представим также, что он может выбрать, через какую сеть (с коммутацией каналов или пакетов) передавать свой трафик, причем в обеих сетях производительность каналов связи одинакова. Очевидно, что более эффективной с точки зрения временны́х затрат для нашего пользователя была бы работа в сети с коммутацией каналов, где ему в единоличное пользование предоставляется зарезервированный канал связи. При этом способе все данные поступали бы адресату без задержки. Тот факт, что значительную часть времени зарезервированный канал будет простаивать (во время пауз), нашего пользователя не волнует — ему важно быстро решить собственную задачу.

Если бы пользователь обратился к услугам сети с коммутацией пакетов, то процесс передачи данных оказался бы более медленным, так как его пакеты, вероятно, не раз задерживались бы в очередях, ожидая освобождения необходимых сетевых ресурсов наравне с пакетами других абонентов.

Давайте рассмотрим подробнее механизм возникновения задержек при передаче данных в сетях обоих типов. Пусть от конечного узла $N1$ отправляется сообщение к конечному узлу $N2$ (рис. 3.12). На пути передачи данных расположены два коммутатора.

В сети с коммутацией каналов данные после задержки, связанной с установлением канала, начинают передаваться на стандартной для канала скорости. Время доставки данных адресату T равно сумме времени распространения сигнала в канале t_{prg} (prg — propagation) и времени передачи сообщения в канал t_{trns} (trns — transmission), называемого также **временем сериализации**.

Наличие коммутаторов в сети с коммутацией каналов никак не влияет на суммарное время прохождения данных через сеть.

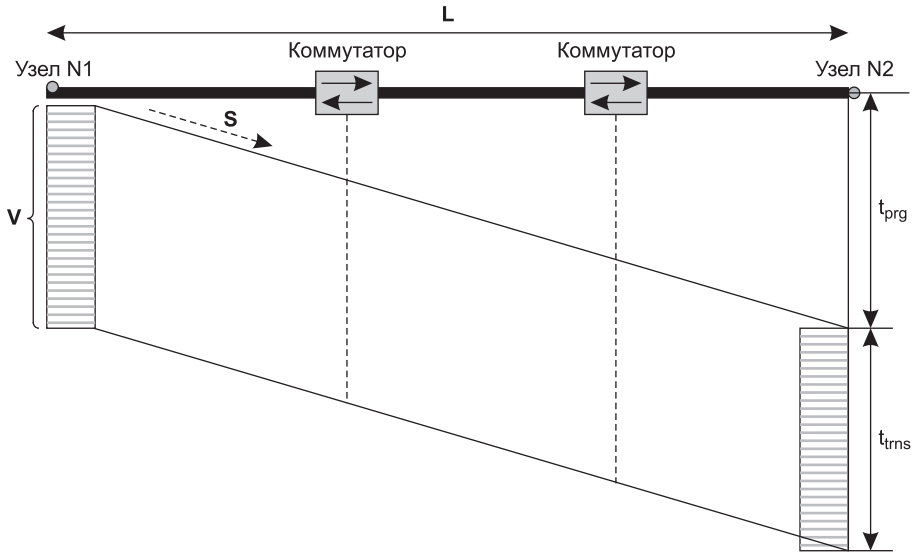


Рис. 3.12. Временна́я диаграмма передачи сообщения в сети с коммутацией каналов

ПРИМЕЧАНИЕ

Заметим, что время передачи сообщения в канал в точности совпадает со временем приема сообщения из канала в буфер узла назначения, то есть временем буферизации.

Время распространения сигнала зависит от расстояния между абонентами L и скорости S распространения электромагнитных волн в конкретной физической среде, которая колеблется от 0,6 до 0,9 скорости света в вакууме:

$$t_{\text{prg}} = L/S.$$

Время сериализации (а значит, и время буферизации в узле назначения) равно отношению объема сообщения V в битах к пропускной способности канала C в битах в секунду:

$$t_{\text{trns}} = V/C, \text{ то есть } T = L/S + V/C.$$

В сети с коммутацией пакетов передача данных не требует обязательного установления соединения. Предположим, что в сеть, показанную на рис. 3.13, передается сообщение того же объема V, что и в предыдущем случае (см. рис. 3.12), однако оно разделено на пакеты, каждый из которых снабжен заголовком. Пакеты передаются от узла $N1$ узлу $N2$, между которыми расположены два коммутатора. На каждом коммутаторе каждый пакет изображен дважды: в момент прихода на входной интерфейс и в момент передачи в сеть с выходного интерфейса. Как видим, коммутатор задерживает пакет на некоторое время. Здесь T_1 — время доставки адресату первого пакета сообщения, а T_{ps} — всего сообщения.

Сравнивая временны́е диаграммы передачи данных в сетях с коммутацией каналов и пакетов, отметим два факта:

- значения времени распространения сигнала (t_{prg}) в одинаковой физической среде на одно и то же расстояние одинаковы;

- учитывая, что значения пропускной способности каналов в обеих сетях одинаковы, значения времени передачи сообщения в канал (t_{trns}) будут *также равны*.

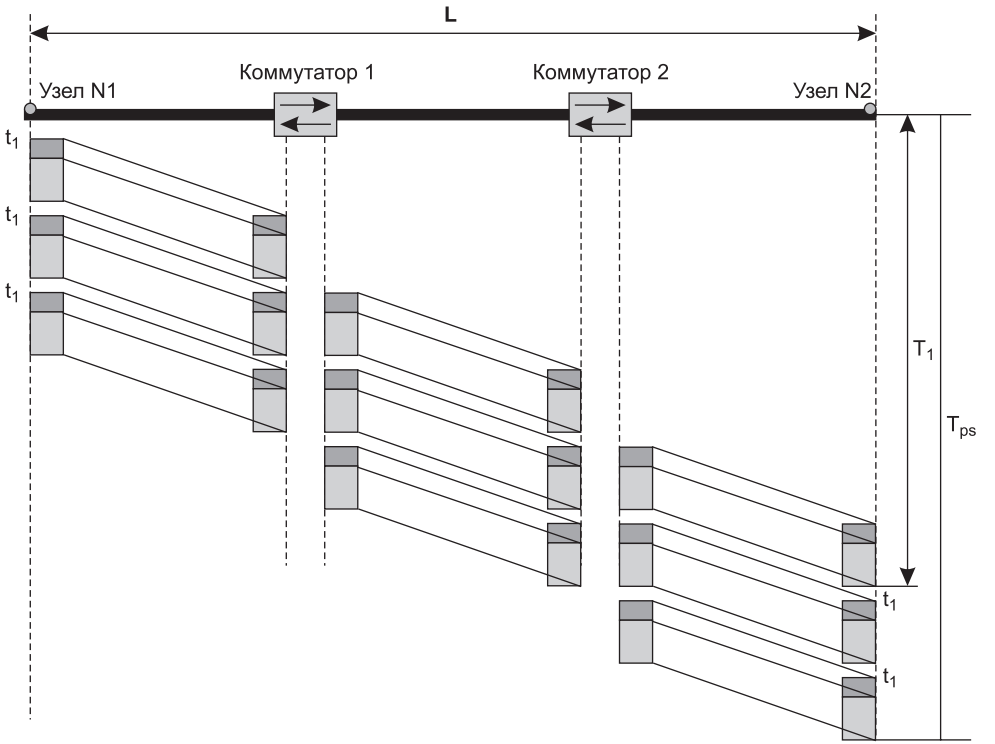


Рис. 3.13. Временна́я диаграмма передачи сообщения, разделенного на пакеты, в сети с коммутацией пакетов

Однако разбиение передаваемого сообщения на пакеты с последующей их передачей по сети с коммутацией пакетов приводит к дополнительным задержкам. Проследим путь первого пакета и отметим, из каких составляющих складывается время его передачи в узел назначения и какие из них специфичны для сети с коммутацией пакетов (рис. 3.14).

Время передачи одного пакета от узла *N*1 до коммутатора 1 можно представить в виде суммы нескольких слагаемых.

- Во-первых, время тратится в узле-отправителе *N*1:
  - t_1 — время формирования пакета, также называемое временем пакетизации (зависит от различных параметров работы программного и аппаратного обеспечения узла-отправителя и не зависит от параметров сети);
  - t_2 — время передачи в канал заголовка;
  - t_3 — время передачи в канал поля данных пакета.
- Во-вторых, дополнительное время тратится на распространение сигналов по каналам связи. Обозначим через t_4 время распространения сигнала, представляющего один бит информации, от узла *N*1 до коммутатора 1.

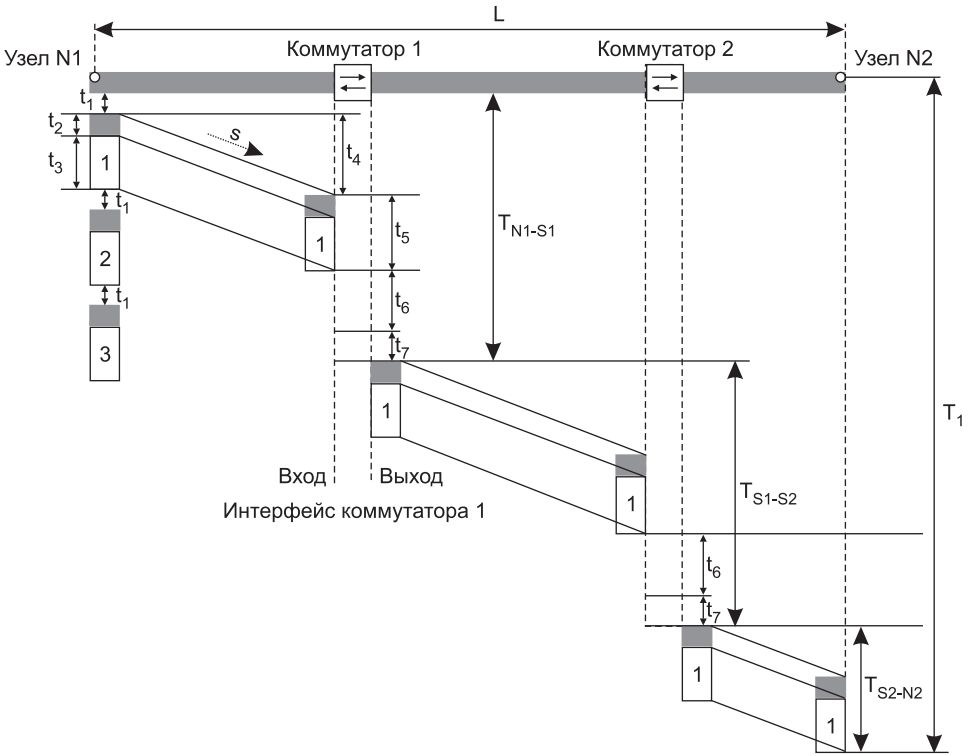


Рис. 3.14. Временная диаграмма передачи одного пакета в сети с коммутацией пакетов

- В-третьих, дополнительное время тратится в промежуточном коммутаторе:
  - t_5 — время приема пакета с его заголовком из канала во входной буфер коммутатора; как отмечено, это время равно ($t_2 + t_3$), то есть времени передачи пакета с заголовком в канал из узла источника;
  - t_6 — время ожидания пакета в очереди, колеблется в очень широких пределах и заранее не известно, так как зависит от текущей загрузки сети;
  - t_7 — время коммутации пакета при его передаче в выходной порт, фиксировано для конкретной модели и обычно невелико (от нескольких микросекунд до нескольких миллисекунд).

Обозначим через T_{N1-S1} время передачи пакета из узла $N1$ на выходной интерфейс коммутатора 1. Это время складывается из следующих составляющих:

$$T_{N1-S1} = t_1 + t_4 + t_5 + t_6 + t_7.$$

Обратите внимание, что среди слагаемых отсутствуют составляющие t_2 и t_3. Из рис. 3.14 видно, что передача битов из передатчика в канал совмещается по времени с передачей битов по каналу связи.

Время, затрачиваемое на оставшиеся два отрезка пути, обозначим соответственно T_{S1-S2} и T_{S2-N2}. Эти величины имеют такую же структуру, что и T_{N1-S1}, за исключением того, что в них не входит время пакетизации, и кроме того, T_{S2-N2} не включает время коммутации

(так как отрезок заканчивается конечным узлом). Итак, полное *время передачи одного пакета* по сети составляет:

$$T_1 = T_{N1-S1} + T_{S1-S2} + T_{S2-N2}.$$

Чему же будет равно время передачи сообщения, состоящего из нескольких пакетов? Сумме времен передачи каждого пакета? Конечно нет! Ведь сеть с коммутацией пакетов работает как конвейер (см. рис. 3.13): пакет обрабатывается в несколько этапов, причем все устройства сети выполняют эти этапы параллельно. Поэтому время передачи такого сообщения будет значительно меньше, чем сумма значений времени передачи каждого пакета сообщения. Точно рассчитать это время сложно из-за неопределенности состояния сети и вследствие этого неопределенности значений времени ожидания пакетов в очередях коммутаторов. Однако если предположить, что пакеты стоят в очереди примерно одинаковое время, то общее *время передачи сообщения, состоящего из n пакетов*, можно оценить следующим образом:

$$T_{PS} = T_1 + (n - 1)\,(t_1 + t_5).$$

Количественное сравнение задержек. Пример

В качестве иллюстрации решим задачу о сравнении скоростей работы сетей с коммутацией каналов и пакетов для частного примера, сделав некоторые предположения о необходимых исходных данных. Пусть, например, нам известно следующее:

- Объем V сообщения, передаваемого в обоих видах сетей, составляет 10^7 байт.
- Отправитель $N1$ находится от получателя $N2$ на расстоянии L = 5000 км.
- Скорость S распространения сигнала составляет 200 000 км/с (2/3 скорости света).
- Пропускная способность линий связи C составляет 100 Мбит/с.

Время передачи данных по сети с коммутацией каналов складывается из времени распространения сигнала и времени передачи сообщения в канал. Будем считать, что канал постоянный, то есть он уже скоммутирован, так что время установления соединения равно нулю.

Время распространения сигнала для расстояния 5000 км можно оценить примерно в 25 мс (5000 км/200 000 км/с = 0,025 с).

Время передачи сообщения в канал при пропускной способности 100 Мбит/с и размере сообщения 10 000 000 байт равно: $(10^7 \times 8$ бит$)/(10^8$ бит/с$) = 8 \times 10^{-1} = 0{,}8$(с) = 800 мс.

То есть передача всех данных абоненту $N2$ в сети с коммутацией каналов занимает 825 мс.

Теперь подсчитаем время передачи такого же объема данных — 10^7 байт — в сети с *коммутацией пакетов*, считая, что:

- Пропускная способность линий связи имеет то же значение — 100 Мбит/с.
- Число промежуточных коммутаторов ($S1$–$S10$) равно десяти.
- Исходное сообщение разбивается на пакеты по 10^3 байт с заголовком 40 байт.
- Интервалы (t_1) между всеми пакетами одинаковы и равны 100 мкс.
- Времена коммутации (t_7) на каждом коммутаторе одинаковы и равны 50 мкс.

$$T_{N1-S1} = 100 \text{ мкс} + t_4^{(1)} + 83 \text{ мкс} + 50 \text{ мкс} = 233 \text{ мкс} + t_4^{(1)}.$$

Время T_{S1-S2} передачи первого пакета от коммутатора $S1$ до коммутатора $S2$ отличается отсутствием составляющей t_1 = 100 мкс (пакет уже сформирован) и значением составляющей t_4, которая зависит от расстояния L_2 между коммутаторами $S1$ и $S1$. Отразим этот факт в обозначении $t_4^{(2)}$.

С учетом сказанного выше имеем T_{S1-S2} = 133 мкс + $t_4^{(2)}$.

Аналогично время передачи данных между каждой парой соседних коммутаторов можно представить в виде: $T_{Si-Si+1}$ = 133 мкс + $t_4^{(i+1)}$, где i изменяется от 1 до 10.

Время T_{S10-N2} передачи первого пакета от коммутатора S10 до конечного узла $N2$ отличается от времени передачи данных между коммутаторами отсутствием составляющей t_7 = 50 мкс (в конечном узле отсутствует операция коммутации) и значением составляющей $t_4^{(11)}$, которая зависит от расстояния $L11$ между коммутатором $S10$ и конечным узлом $N2$.

С учетом сказанного выше имеем T_{S10-N2} = 83 мкс + $t_4^{(11)}$.

Теперь найдем суммарное время T_1 прохождения первого пакета от узла N1 до узла N2.

$$T_1 = T_{N1-S1} + T_{S1-S2} + ... + T_{S10-N2} = 233 \text{ мкс} + (133 \text{ мкс}) \times 9 + 83 \text{ мкс} + \Sigma\, t_4^{(i)} = \\ = 1513 \text{ мкс} + (\Sigma\, L_i)/S,$$

где i изменяется от 1 до 11. Сумма расстояний между всеми соседними узлами и коммутаторами равна расстоянию L между источником и приемником, то есть те же 5000 км, что и в сети с коммутацией каналов. Отсюда получаем полное время передачи первого пакета:

$$T_1 = T_{N1-N2} = 0{,}001513 \text{ с} + (5000 \text{ км})/(200\,000 \text{ км/с}) = 0{,}001513 \text{ с} + 0{,}025 \text{ с} = 0{,}026513 \text{ с}.$$

Каждый из следующих пакетов будет прибывать через интервал 100 мкс и занимать еще 83 мкс на буферизацию. Отсюда время передачи сообщения, состоящего из 10 000 пакетов, можно оценить следующим образом:

$$T_{PS} = T_1 + (10\,000 - 1)\,(t_1 + t_5) = 0{,}026513 + 9999 \times (0{,}0001 + 0{,}000083) = 0{,}026513 + \\ + 9999 \times 0{,}000183 = 0{,}026513 + 1{,}829817 = 1{,}85633 \text{ с} = 1856 \text{ мс}.$$

Таким образом, передача данных при указанных условиях по сети с коммутацией пакетов займет 1856 мс, что на 1031 мс дольше, чем в сети с коммутацией каналов, в которой эта же передача занимает 825 мс.

Ethernet — пример стандартной технологии с коммутацией пакетов

Рассмотрим, каким образом описанные ранее концепции воплощены в одной из первых стандартных сетевых технологий — технологии Ethernet, работающей со скоростью 10 Мбит/с. В этом разделе мы коснемся только самых общих принципов функционирования Ethernet (детальное описание технологии Ethernet приведено в части III этой книги).

- *Топология.* Существует два варианта технологии Ethernet: Ethernet на разделяемой среде и коммутируемый вариант Ethernet. В первом случае все узлы сети разделяют общую среду передачи данных, то есть сеть строится по топологии общей шины. На рис. 3.15 показан простейший вариант топологии — все компьютеры сети подключены к общей

разделяемой среде, состоящей из одного сегмента коаксиального кабеля, который в данном случае является полудуплексным каналом связи.

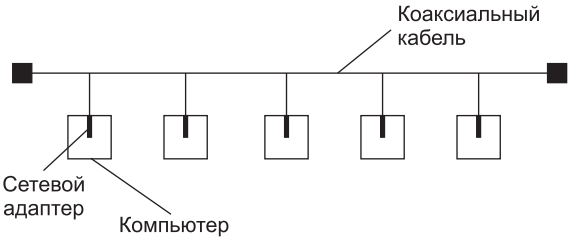


Рис. 3.15. Сеть Ethernet на разделяемой среде

В том случае, когда сеть Ethernet не использует разделяемую среду, а строится на коммутаторах, объединенных дуплексными каналами связи, говорят о коммутируемом варианте Ethernet. Топология в этом случае является топологией дерева, то есть такой, при которой между двумя любыми узлами сети существует ровно один путь. Пример топологии коммутируемой сети Ethernet показан на рис. 3.16.

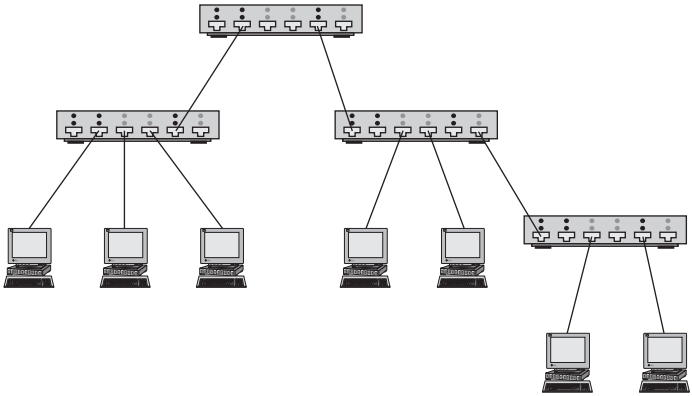


Рис. 3.16. Древовидная топология коммутируемой сети Ethernet

Топологические ограничения (только древовидная структура связей коммутаторов) связаны со способом построения таблиц продвижения Ethernet-коммутаторами.

- *Способ коммутации.* В технологии Ethernet используется дейтаграммная коммутация пакетов. Единицы данных, которыми обмениваются компьютеры в сети Ethernet, называются кадрами. Кадр имеет фиксированный формат и наряду с полем данных содержит различную служебную информацию. В том случае, когда сеть Ethernet построена на коммутаторах (см. рис. 3.16), каждый коммутатор продвигает кадры в соответствии с теми принципами коммутации пакетов, которые были описаны ранее. А вот в случае односегментной сети Ethernet на разделяемой среде (см. рис. 3.15) возникает законный вопрос: где же выполняется коммутация? Где хотя бы один коммутатор, который, как мы сказали, является главным элементом любой сети с коммутацией пакетов? Или же Ethernet поддерживает особый вид коммутации? Оказывается, «коммутатор» в односегментной сети Ethernet существует, но его не так просто разглядеть, потому что его

функции распределены по всей сети между сетевыми адаптерами компьютеров и собственно разделяемой средой передачи сигналов. Интерфейсами этого виртуального коммутатора являются сетевые адаптеры, а функцию коммутационного блока — передачу кадров между интерфейсами — выполняет разделяемая среда. Адаптеры берут на себя и часть функций коммутационного блока: именно они решают, какой кадр адресован их компьютеру, а какой — нет.

- *Адресация.* Каждый компьютер, или, точнее, каждый сетевой адаптер, имеет уникальный аппаратный адрес (так называемый MAC-адрес, вы уже встречали этот акроним в главе 2). Ethernet-адрес является плоским числовым адресом. Поддерживаются адреса для индивидуальной, широковещательной и групповой рассылки.
- *Разделение среды и мультиплексирование.* В сети Ethernet на коммутаторах каждый канал является дуплексным каналом связи, вследствие чего проблемы его разделения между интерфейсами узлов не возникает. Передатчики Ethernet-коммутаторов используют дуплексные каналы связи для мультиплексирования потоков кадров от разных конечных узлов.

  В случае Ethernet на разделяемой среде конечные узлы применяют специальный метод доступа с целью синхронизации использования единственного полудуплексного канала связи, объединяющего все компьютеры сети. Единого арбитра в сети Ethernet на разделяемой среде нет, вместо этого все узлы прибегают к распределенному случайному методу доступа. Информационные потоки, поступающие от конечных узлов сети Ethernet, мультиплексируются в единственном передающем канале в режиме разделения времени. То есть кадрам разных потоков поочередно предоставляется канал. Чтобы подчеркнуть не всегда очевидную разницу между понятиями мультиплексирования и разделения среды, рассмотрим ситуацию, когда из всех компьютеров сети Ethernet только одному нужно передавать данные, причем данные нескольких приложений. В этом случае проблема разделения среды между сетевыми интерфейсами не возникает, в то время как задача передачи нескольких информационных потоков по общей линии связи (то есть мультиплексирование) остается.
- *Кодирование.* Адаптеры в Ethernet работают с тактовой частотой 20 МГц, передавая в среду прямоугольные импульсы, соответствующие единицам и нулям данных компьютера. Когда начинается передача кадра, все его биты передаются в сеть с постоянной скоростью 10 Мбит/с (каждый бит передается за два такта). Эта скорость определяет пропускную способность линии связи в сети Ethernet.
- *Надежность.* Для повышения надежности передачи данных в Ethernet используется стандартный прием — подсчет **контрольной суммы** и передача ее в концевике кадра. Если принимающий адаптер путем повторного подсчета контрольной суммы обнаруживает ошибку в данных кадра, то такой кадр отбрасывается. Повторная передача кадра протоколом Ethernet не выполняется — эта задача должна решаться средствами более высокого уровня, например протоколом TCP в сетях TCP/IP.
- *Очереди.* В те периоды времени, когда среда занята передачей кадров других сетевых адаптеров, данные приложений (предложенная нагрузка) по-прежнему поступают в сетевой адаптер. Так как они не могут быть переданы в это время в сеть, то начинают накапливаться во внутреннем буфере Ethernet-адаптера, образуя очередь. Поэтому в сети Ethernet, как и во всех сетях с коммутацией пакетов, существуют переменные задержки доставки кадров.

ГЛАВА 4 Стандартизация и классификация сетей

Декомпозиция задачи сетевого взаимодействия

Сетевая архитектура — это концептуальная схема функционирования компьютерной сети, определяющая принципы работы аппаратных и программных сетевых компонентов, организацию их связей, протоколы взаимодействия и способы физической передачи данных. Архитектура сети отражает декомпозицию общей задачи взаимодействия компьютеров на отдельные подзадачи, которые должны решаться отдельными компонентами сети — конечными узлами (компьютерами) и промежуточными узлами (коммутаторами и маршрутизаторами).

Многоуровневый подход

Для решения сложных задач, к которым относится и задача сетевого взаимодействия, используется известный универсальный прием — *декомпозиция*, то есть разбиение одной сложной задачи на несколько более простых задач-модулей. Декомпозиция состоит в четком определении функций каждого модуля, а также порядка их взаимодействия (то есть межмодульных интерфейсов). При таком подходе каждый модуль можно рассматривать как «черный ящик», абстрагируясь от его внутренних механизмов и концентрируя внимание на способе взаимодействия модулей. В результате такого логического упрощения задачи появляется возможность независимого тестирования, разработки и модификации модулей. Так, любой из показанных на рис. 4.1 модулей может быть переписан заново. Пусть, например, это будет модуль *A*, и если при этом разработчики сохранят без изменения межмодульные связи (в данном случае — интерфейсы *A*—*B* и *A*—*C*), то это не потребует никаких изменений в остальных модулях.

Еще более эффективной концепцией, развивающей идею декомпозиции, является *многоуровневый подход*. После представления исходной задачи в виде множества модулей эти модули группируют и упорядочивают по уровням, образующим иерархию. В соответствии с принципом иерархии для каждого промежуточного уровня можно указать непосредственно примыкающие к нему соседние вышележащий и нижележащий уровни (рис. 4.2).

С одной стороны, группа модулей, составляющих каждый уровень, для решения своих задач должна обращаться с запросами только к модулям соседнего нижележащего уровня. С другой стороны, результаты работы каждого из модулей, отнесенных к некоторому уровню, могут быть переданы только модулям соседнего вышележащего уровня. Такая иерархическая декомпозиция задачи предполагает четкое определение функций и интерфейсов не только отдельных модулей, но и уровней в целом.

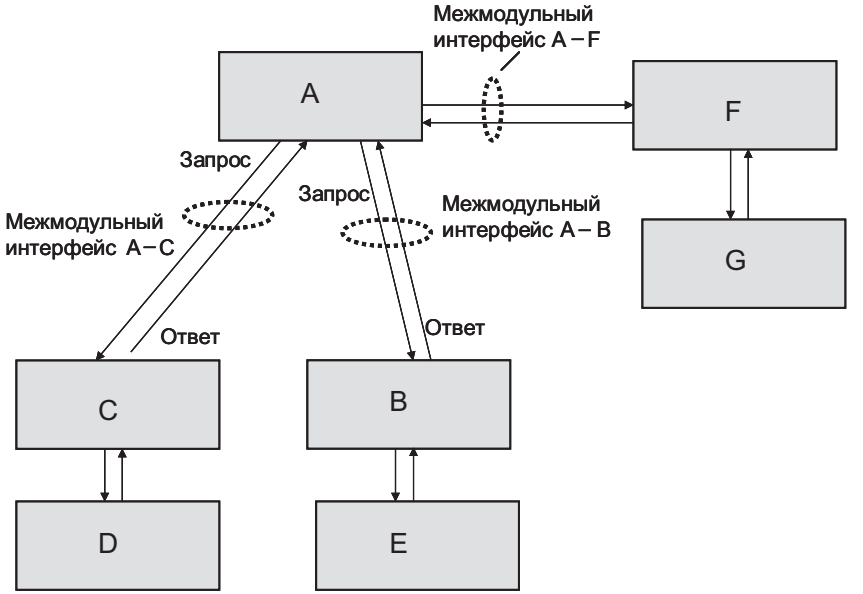


Рис. 4.1. Пример декомпозиции задачи

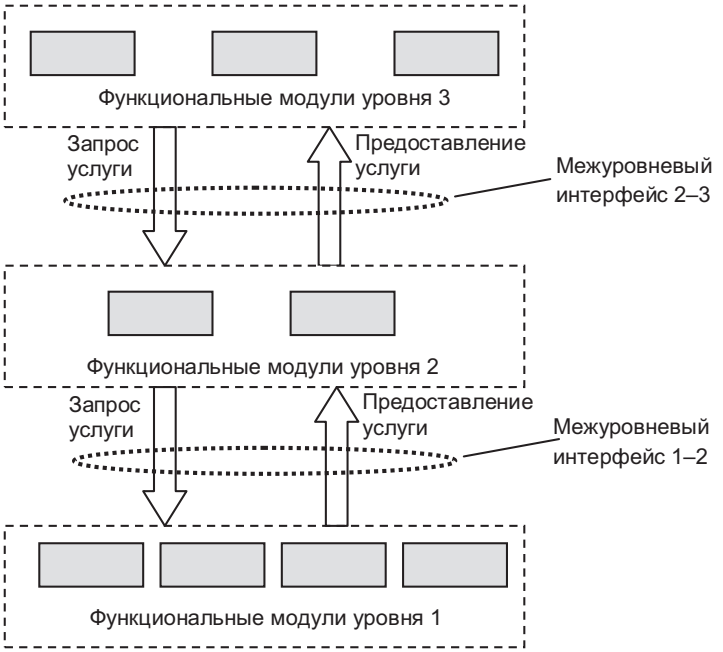


Рис. 4.2. Многоуровневый подход — создание иерархии задач

Межуровневый интерфейс, называемый также **интерфейсом услуг**, определяет набор функций, которые нижележащий уровень предоставляет вышележащему (рис. 4.3).

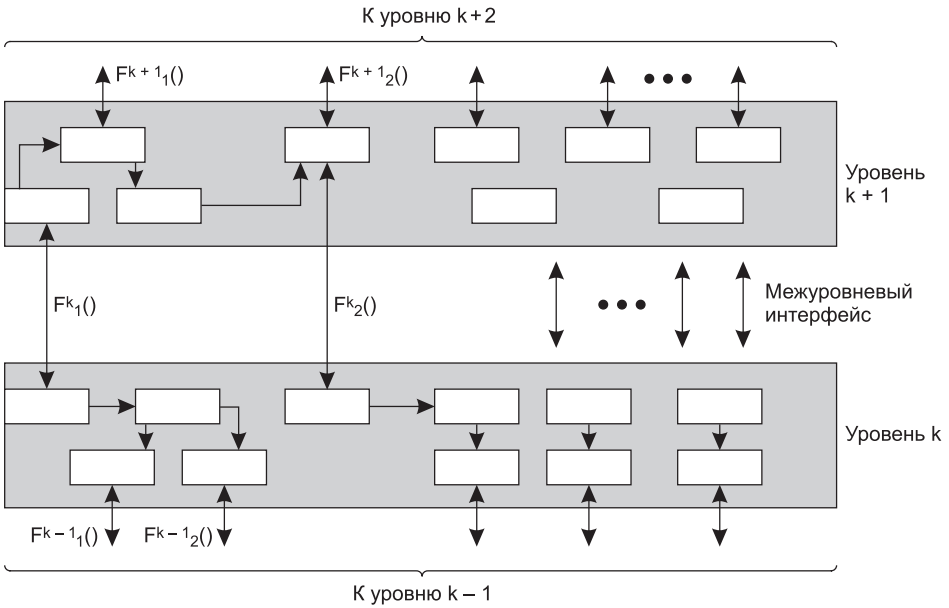


Рис. 4.3. Концепция многоуровневого взаимодействия

Такой подход дает возможность проводить разработку, тестирование и модификацию каждого отдельного уровня независимо от других уровней. Иерархическая декомпозиция позволяет, двигаясь от более низкого уровня к более высокому, переходить ко все более и более абстрактному, а значит, более простому представлению исходной задачи.

Протокол и стек протоколов

Многоуровневое представление средств сетевого взаимодействия имеет свою специфику, связанную с тем, что в процессе обмена сообщениями участвуют, по меньшей мере, *две стороны*, то есть в данном случае необходимо организовать согласованную работу двух иерархий аппаратных и программных средств на разных компьютерах. Оба участника сетевого обмена должны принять множество соглашений. Например, они должны согласовать параметры электрических сигналов, длину сообщений, договориться о методах контроля достоверности и т. п. Другими словами, соглашения должны быть приняты на всех уровнях, начиная от самого низкого — уровня передачи битов — и заканчивая самым высоким, реализующим обслуживание пользователей сети.

На рис. 4.4 показана модель взаимодействия двух узлов. С каждой стороны средства взаимодействия представлены четырьмя уровнями. Каждый уровень поддерживает интерфейсы двух типов. Во-первых, это интерфейсы услуг с выше- и нижележащим уровнем «своей» иерархии средств. Во-вторых, это *одноранговый* интерфейс со средствами взаимодействия другой стороны, расположенными на том же уровне иерархии. Этот тип интерфейса называют **протоколом**.

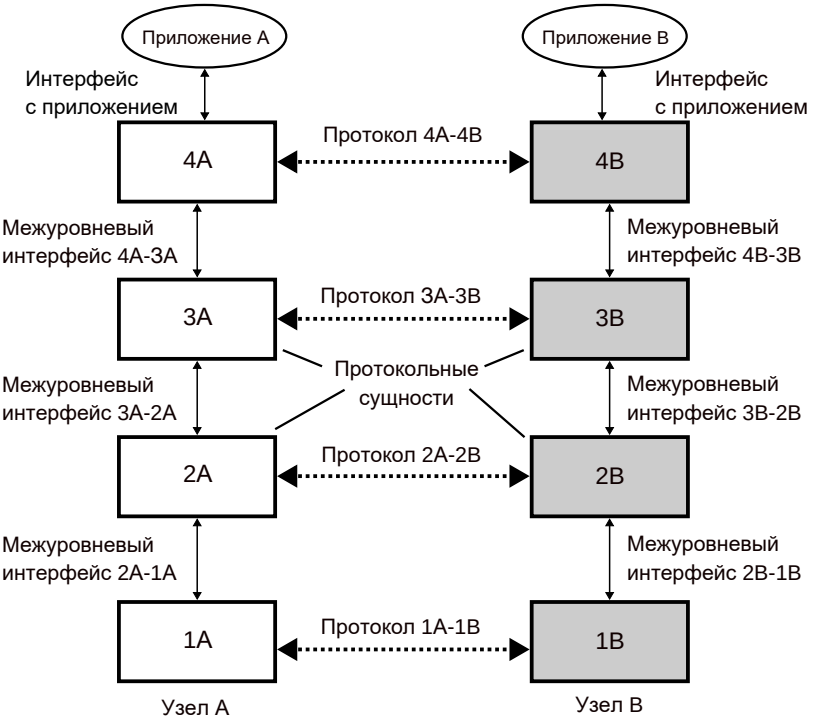


Рис. 4.4. Взаимодействие двух узлов

В сущности, термины «протокол» и «интерфейс» обозначают одно и то же — формализованное описание процедуры взаимодействия двух объектов, но традиционно в сетях за ними закрепили разные области предпочтительного использования: протоколы определяют правила взаимодействия модулей одного уровня (одного ранга) в разных узлах, а интерфейсы — правила взаимодействия модулей соседних уровней в одном узле.

Иерархически организованный набор протоколов, достаточный для организации взаимодействия узлов в сети, называется **стеком протоколов**.

Протокол включает в себя не только набор сообщений, которыми обмениваются два узла на определенном уровне взаимодействия, но и описание логики обработки этих сообщений узлом при их получении. Протокол регулирует правила передачи и получения данных между двумя или более устройствами, обеспечивая их эффективное взаимодействие. Протокольные соглашения определяют формат, синхронизацию и последовательность обмена данными, а также порядок обработки ошибок. Все сетевые устройства общаются друг с другом посредством сетевых протоколов.

Протоколы нижних уровней часто реализуются комбинацией программных и аппаратных средств, а протоколы верхних уровней, как правило, — программными средствами. Программный или программно-аппаратный модуль, реализующий некоторый протокол, называют **протокольной сущностью** или, для краткости, тоже протоколом. Понятно, что

один и тот же протокол может быть реализован с разной степенью эффективности. Именно поэтому при сравнении протоколов следует учитывать не только логику их работы, но и *качество реализации*. Более того, на эффективность взаимодействия устройств в сети влияет качество всей совокупности протоколов, составляющих стек, — в частности то, *насколько рационально распределены функции между протоколами* разных уровней и насколько хорошо определены интерфейсы между ними.

Протокольные сущности одного уровня обмениваются **сообщениями** в соответствии с определенным для них протоколом. Сообщения состоят из заголовка и поля данных (иногда оно может отсутствовать). Работа каждого протокольного модуля состоит в интерпретации заголовков поступающих к нему сообщений и выполнении связанных с этим действий. Заголовки сообщений разных протоколов имеют разную структуру, что соответствует различиям в их функциональности.

Модель OSI

Цель создания модели OSI

В начале 80-х годов ряд международных организаций по стандартизации, в частности International Standards Organization (ISO), а также International Telecommunications Union (ITU) и некоторые другие разработали стандартную модель **взаимодействия открытых систем** (Open System Interconnection, OSI). Эта модель сыграла значительную роль в развитии компьютерных сетей.

К концу 70-х годов в мире уже существовало большое количество активно используемых стеков коммуникационных протоколов — таких, например, как DECnet, TCP/IP и IBM SNA. Каждому из них соответствовала собственная модель разбиения общей задачи сетевого взаимодействия на иерархически организованные уровни. (Мы называем многоуровневое представление **моделью** в том случае, когда задачи сетевого взаимодействия каждого уровня описаны максимально укрупненно, в самом общем виде — виде перечня функций.)

Разнообразие средств межсетевого взаимодействия вывело на первый план проблему *несовместимости* устройств, использующих разные протоколы. Одним из путей разрешения этой проблемы в то время виделся всеобщий переход на единый, общий для всех систем стек протоколов, созданный с учетом недостатков уже существующих стеков. Поэтому было решено разработать универсальную модель OSI, которая должна была служить *основой* для создания нового универсального стандартного стека протоколов для сетей с коммутацией пакетов.

По замыслу разработчиков, модель должна была представлять разбиение обобщенной задачи сетевого взаимодействия на несколько иерархических уровней и перечень задач, решаемых на каждом уровне. Этих уровней не должно быть слишком много, чтобы процедуры взаимодействия между ними не приводили к слишком большим накладным расходам. Но уровней должно быть достаточно, чтобы задачи каждого из них могли быть четко определены. Каждый уровень должен предоставлять определенный набор услуг следующему, более высокому уровню, при этом нужно стремиться к максимальной обособленности уровней. Имея такую модель, можно вместо подробного описания функций того или иного протокола или сетевого устройства просто сказать, к какому уровню модели OSI относится этот протокол или устройство.

Однако такая задача оказалось непростой, потому что декомпозиция реальных задач на независимые уровни не всегда имеет решение, так как связи между задачами могут быть слишком сложными. Например, решение некоторой задачи на определенном уровне может потребовать услуг не непосредственно лежащего под ним уровня, а более низкого. Разработка модели OSI заняла большой период времени, с 1977 по 1984 год. Если измерять ее успех созданием стека протоколов, соответствующего этой модели, то нужно признать, что успех этот оказался частичным. Единственным стеком протоколов, структура которого полностью соответствовала модели OSI, оказался стек протоколов OSI, созданный инициаторами этого проекта. Однако и он не стал востребованным и поэтому не стал универсальным.

С другой стороны, создание *модели* OSI можно признать успехом, так как ее архитектура действительно стала общим знаменателем различных протоколов и их стеков и при описании назначения какого-либо протокола достаточно упомянуть, к какому уровню модели OSI он относится.

Определение уровней модели OSI

Содержание модели OSI является весьма компактным: она стандартизует (1) количество уровней, а именно 7, (2) названия уровней и (3) функции уровней.

Уровни модели OSI:

- прикладной (application layer);
- представления (presentation layer);
- сеансовый (session layer);
- транспортный (transport layer);
- сетевой (network layer);
- канальный (data link layer);
- физический (physical layer).

Физический уровень модели OSI имеет дело с передачей потока битов по физическим каналам связи, таким как коаксиальный кабель, витая пара, оптоволоконный кабель или беспроводная линия связи. Этот уровень обеспечивает представление потока битов в виде сигналов определенной формы — электрических или оптических. Тем самым этот уровень определяет электромеханический интерфейс (параметры сигналов и разъемов) между сетевыми устройствами. Физический уровень не распознает структуру информации, которую он передает, не делит ее на какие-либо информационные единицы, не проверяет ошибок.

Канальный уровень, используя возможности, предоставляемые ему нижележащим, физическим, уровнем, предлагает вышележащему, сетевому, уровню следующие услуги:

- установление логического соединения между двумя непосредственно связанными узлами;
- согласование в рамках соединения скоростей передатчика и приемника информации;
- обеспечение надежной передачи, обнаружение и коррекцию ошибок.

Протокольной единицей данных канального уровня является **кадр**. В поле данных кадра размещается сообщение сетевого уровня. Для обнаружения ошибок при передаче кадров канальный уровень использует такие средства, как контрольная сумма поля данных кадра.

Для коррекции ошибок предполагается повторная передача кадра с искаженной информацией. В функции канального уровня входит адресация узлов.

Функции канального и физического уровней обычно реализуются на *всех* устройствах, подключенных к сети. Со стороны компьютера эти функции выполняются сетевым адаптером, со стороны промежуточных сетевых устройств — коммутаторов, маршрутизаторов, мультиплексоров и других — входными и выходными интерфейсами (портами).

Сетевой уровень. Теперь, когда определена функция канального уровня, состоящая в решении задачи надежной передачи кадров между двумя *соседними* узлами сети, можно поставить задачу более высокого уровня — передача данных между *любыми* узлами сети, в общем случае не связанными друг с другом.

Сетевой уровень отвечает за передачу данных между двумя конечными узлами, не обязательно непосредственно связанными друг с другом (возможно, находящимися в разных сетях). Для этого он решает две задачи:

- **продвижение** данных между заданными конечными узлами сети на основании адресов сетевого уровня;
- **нахождение маршрута** между этими конечными узлами через промежуточные узлы сети.

Очевидно, что организация связи двух любых, не обязательно соседних, узлов является более общей, и поэтому ее решение может быть достигнуто путем многократных обращений к нижележащему уровню. Так, организация взаимодействия узлов *A* и *B* может быть сведена к поочередному взаимодействию пар промежуточных смежных узлов (рис. 4.5).

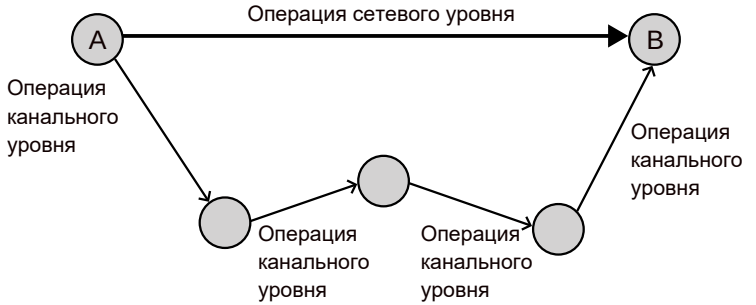


Рис. 4.5. Взаимодействие произвольной пары узлов

В общем случае функции сетевого уровня шире, чем определение маршрутов и обеспечение обмена в пределах составной сети. Например, к сетевому уровню отнесена задача создания надежных и гибких барьеров на пути нежелательного трафика между сетями.

Транспортный уровень обеспечивает приложениям и верхним уровням стека — прикладному, представления и сеансовому — передачу данных с той степенью надежности, которая им требуется. Модель OSI определяет пять **классов транспортного сервиса**, от низшего класса 0 до высшего класса 4. Эти виды сервиса отличаются качеством предоставляемых услуг: срочностью, возможностью восстановления прерванной связи, наличием средств мультиплексирования нескольких соединений между различными прикладными протоколами через общий транспортный протокол, а главное — способностью к обнаружению и исправлению ошибок передачи, таких как искажение, потеря и дублирование пакетов, которые не смогли исправить нижние уровни.

Сеансовый уровень управляет взаимодействием сторон: фиксирует, какая из сторон является активной в настоящий момент, и предоставляет средства синхронизации сеанса. Эти средства позволяют в ходе длинных передач сохранять информацию о состоянии этих передач в виде контрольных точек, чтобы в случае отказа можно было вернуться назад к последней контрольной точке, а не начинать все с начала. На практике немногие приложения используют сеансовый уровень, который редко реализуется в виде отдельных протоколов. Функции этого уровня часто объединяют с функциями прикладного уровня и реализуют в одном протоколе.

Уровень представления, как следует из его названия, обеспечивает представление передаваемой по сети информации, не меняя при этом ее содержания. За счет уровня представления информация, передаваемая прикладным уровнем одной системы, всегда понятна прикладному уровню другой системы. С помощью средств данного уровня протоколы прикладных уровней могут преодолеть синтаксические различия в представлении данных или же различия в кодах символов, например кодов ASCII и EBCDIC. На этом уровне могут выполняться шифрование и дешифрование данных, благодаря которым секретность обмена данными обеспечивается сразу для всех прикладных служб.

Прикладной уровень модели OSI отвечает за предоставление разнообразных услуг пользовательским приложениям — таких, как доступ к общим сетевым ресурсам (файлам, принтерам или веб-страницам) или распределенным сетевым сервисам (электронной почте, службам передачи сообщений, базам данных). Как правило, услуги прикладного уровня включают идентификацию и аутентификацию участников сетевого взаимодействия, проверку их доступности и полномочий, определение требований к защищенности сеанса обмена и т. д. Для запросов к прикладному уровню используются системные вызовы операционной системы, образующие прикладной программный интерфейс. Операционная система выполняет процедуры доступа к услугам прикладного уровня прозрачным для приложений образом, экранируя их от всех деталей устройства транспортной подсистемы сети, а также работы сеансового и представительского уровней.

> В стандартах ISO для обозначения единиц обмена данными, с которыми имеют дело протоколы разных уровней, используется общее название **протокольная единица данных** (Protocol Data Unit, PDU). Для обозначения единиц обмена данными конкретных уровней часто используются специальные названия, в частности: **сообщение**, **кадр**, **пакет**, **дейтаграмма**, **сегмент**.

В заключение подчеркнем разницу между моделью, стеками протоколов и программно-аппаратными реализациями протоколов (рис. 4.6).

Модель говорит о том, **что** должны делать протоколы каждого уровня в максимально обобщенном виде. Протокол говорит о том, **как** нужно это сделать, причем говорит это достаточно детально для однозначного толкования алгоритма работы программы или устройства, реализующих этот протокол. При этом, для каждого уровня модели может существовать несколько протоколов, по-разному выполняющих обобщенно сформулированные в модели задачи этого уровня. И наконец, для каждого протокола могут существовать различные его реализации. При этом проверка соответствия реализации протокола его описанию является важной задачей, для решения которой создаются специальные органы. Например, Wi-Fi Alliance занимается сертификацией продуктов Wi-Fi на соответствие их стандартному протоколу и в случае обнаружения несоответствия запрещает использовать название Wi-Fi.

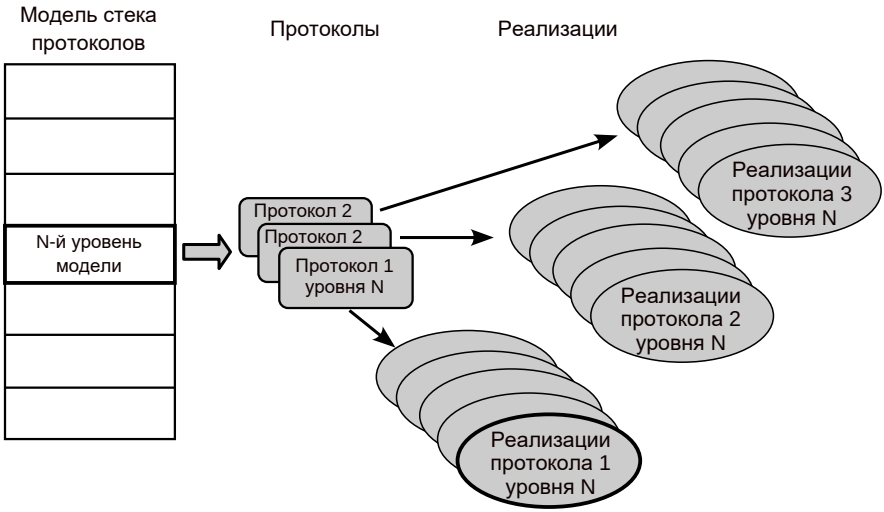


Рис. 4.6. Модель, протокол, реализация протокола

Схема многоуровневого взаимодействия конечных узлов

Рассмотрим упрощенную схему взаимодействия двух сетевых приложений, работающих на компьютерах, на которых установлены стеки протоколов, соответствующие модели OSI (рис. 4.7). Пусть, например, приложение *A* собирает данные, на основе которых приложение *B* генерирует отчет. Совместная работа этих двух приложений обеспечивается за счет обмена сообщениями согласно протоколу, разработанному для них прикладными программистами. Протокольные сообщения отражают логику работы приложения, они могут быть самыми разнообразными, например: «Данные для отчета N готовы», «Несовпадение типа данных» и т. п.

Разработчики приложений *A* и *B* решили, что протокольные сообщения будут передаваться по сети в виде файлов. Пусть, например, приложение *A* посылает сообщение «Данные для отчета N готовы» приложению *B*. Для этого оно, используя интерфейс API, выполняет запрос к системным средствам *прикладного уровня* модели OSI (в нашем примере к сетевой файловой системе) с просьбой передать файл с этим сообщением на удаленный компьютер 2.

В ходе выполнения запроса приложения *A* клиент и сервер файловой системы в соответствии со своим протоколом обмениваются собственной последовательностью команд — протокольных сообщений. В поле данных одной из таких команд клиента файловой системы упаковано сообщение приложения *A*, а в заголовке (на рис. 4.7 заголовок прикладного уровня 7) — служебная информация для сервера файловой системы. Чтобы это сообщение прикладного уровня было доставлено по назначению, необходимо решить еще ряд задач, ответственность за которые несут нижележащие уровни средств сетевого взаимодействия.

Следующим уровнем является *уровень представления*. Протокольный модуль прикладного уровня посредством межуровневого интерфейса передает ему свое сообщение, а также запрос на выполнение тех или иных услуг, предоставляемых этим уровнем. Средства уровня представления выполняют требуемые действия (например, перекодировку информации)

и формируют свое сообщение, добавляя к полученному сообщению собственную служебную информацию (на рис. 4.7 заголовок уровня представления 6).

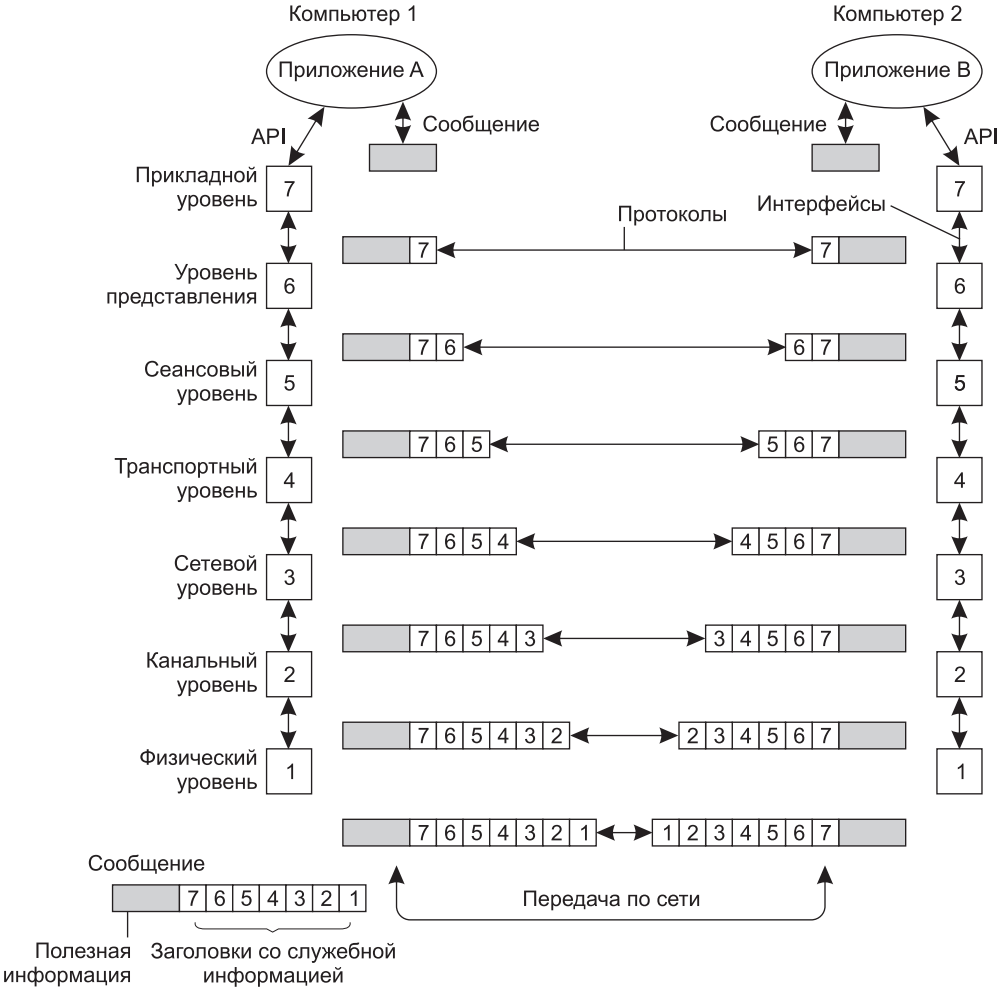


Рис. 4.7. Модель взаимодействия открытых систем ISO/OSI

Сформированное уровнем представления сообщение передается вниз по стеку средствам *сеансового уровня*, которые, в свою очередь, после выполнения функций, определенных для этого уровня, передают свое сообщение средствам *транспортного, сетевого,* а затем *канального* уровня.

Наконец, сообщение достигает средств нижнего, *физического* уровня, которые, собственно, и передают его по линиям связи компьютеру-адресату. К этому моменту сообщение «обрастает» заголовками всех уровней[1].

[1] Некоторые реализации протоколов помещают служебную информацию не только в начале сообщения в виде заголовка, но и в конце (в виде так называемого концевика).

Средства физического уровня помещают сообщение на физический выходной интерфейс компьютера 1, и оно начинает свое «путешествие» по сети (до этого момента сообщение передавалось от одного уровня к другому в пределах компьютера 1).

Когда сообщение по сети поступает на входной интерфейс компьютера 2, оно принимается его физическим уровнем и последовательно перемещается вверх с уровня на уровень. Средства каждого уровня анализируют и обрабатывают заголовок своего уровня, выполняя соответствующие функции, а затем удаляют этот заголовок и передают сообщение вышележащему уровню до тех пор, пока оно не поступит на вход приложения *B*. Приложение *B* соответствующим образом интерпретирует сообщение «Данные для отчета N готовы», например, подготавливает буфер для приема данных.

> Как видно из описания, протокольные сущности одного уровня не общаются между собой непосредственно, в этом общении всегда участвуют посредники — средства протоколов нижележащих уровней. И только физические уровни различных узлов взаимодействуют непосредственно.

Протоколы нижних четырех уровней обобщенно называют сетевым транспортом, или ***транспортной подсистемой***, так как они полностью решают задачу транспортировки сообщений с заданным уровнем качества в составных сетях с произвольной топологией и различными технологиями. Оставшиеся три верхних уровня объединяет то, что они ***тесно связаны с пользовательскими приложениями***, предоставляя им высокоуровневые услуги по использованию сетевых ресурсов и сервисов.

Распределение протоколов по элементам сети

На рис. 4.8 показаны основные элементы компьютерной сети: конечные узлы — компьютеры; коммуникационные узлы — коммутаторы и маршрутизаторы.

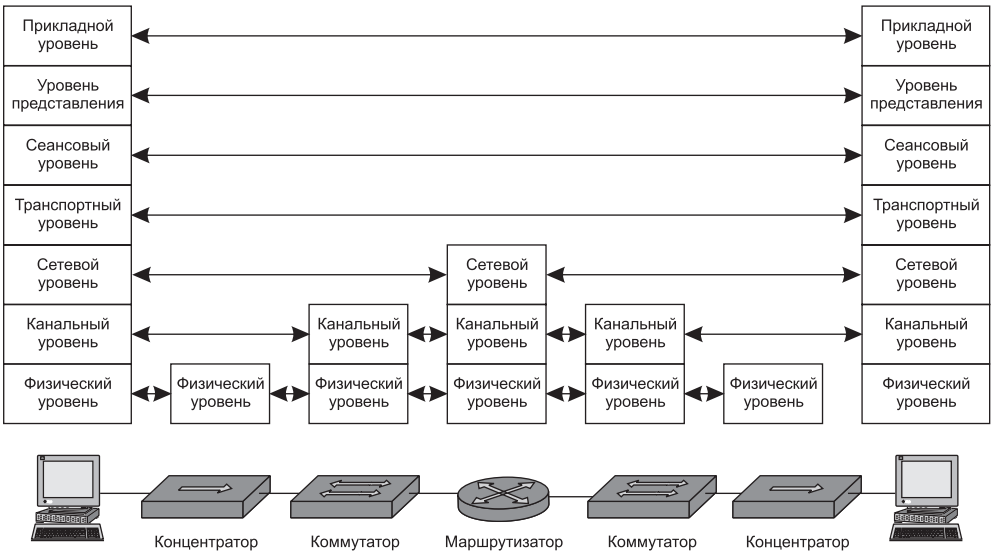


Рис. 4.8. Соответствие функций различных устройств сети уровням модели OSI

Из рисунка видно, что полный стек протоколов реализован только на конечных узлах, а коммуникационным устройствам для продвижения пакетов достаточно функциональности нижних трех уровней. Более того, коммуникационное устройство может поддерживать только протоколы двух нижних уровней или даже одного физического уровня — это зависит от типа устройства.

Именно к таким устройствам, работающим на физическом уровне, относятся, например, сетевые *повторители*, называемые также *концентраторами* или *хабами*. Они повторяют электрические сигналы, поступающие на одни их интерфейсы, на других своих интерфейсах, улучшая характеристики сигналов — мощность и форму, синхронность их следования.

Коммутаторы локальных сетей поддерживают протоколы двух нижних уровней, физического и канального, что дает им возможность работать в пределах стандартных топологий.

Маршрутизаторы должны поддерживать протоколы всех трех уровней, так как сетевой уровень нужен им для объединения сетей различных технологий, а протоколы нижних уровней — для взаимодействия с конкретными сетями, образующими составную сеть, например Ethernet или Frame Relay.

Коммутаторы глобальных сетей (например, MPLS), работающие на основе технологии виртуальных каналов, могут поддерживать как два уровня протоколов, так и три. Протокол сетевого уровня нужен им в том случае, если они поддерживают процедуры автоматического установления виртуальных каналов. Так как топология глобальных сетей произвольна, без сетевого протокола обойтись нельзя. Если же виртуальные соединения устанавливаются администраторами сети вручную, то коммутатору глобальной сети достаточно поддерживать только протоколы физического и канального уровней, чтобы передавать данные по уже проложенным виртуальным каналам.

Компьютеры, на которых работают сетевые приложения, должны поддерживать протоколы всех уровней. Протоколы прикладного уровня, пользуясь сервисами протоколов уровня представления и сеансового уровня, предоставляют приложениям набор сетевых услуг в виде сетевого прикладного программного интерфейса (API). Протокол транспортного уровня также работает на всех конечных узлах. При передаче данных через сеть два модуля транспортного протокола, работающие на узле-отправителе и узле-получателе, взаимодействуют друг с другом для поддержания транспортного сервиса нужного качества. Коммуникационные устройства сети переносят сообщения транспортного протокола прозрачным образом, не вникая в их содержание.

Вспомогательные протоколы

На рис. 4.8 показан упрощенный вариант распределения протоколов между элементами сети. В реальных сетях некоторые из коммуникационных устройств поддерживают не только протоколы трех нижних уровней, но и протоколы верхних уровней. Так, маршрутизаторы реализуют протоколы маршрутизации, позволяющие автоматически строить таблицы маршрутизации, а концентраторы и коммутаторы часто поддерживают протоколы SNMP и telnet, которые не нужны для выполнения основных функций этих устройств, но позволяют конфигурировать их и управлять ими удаленно. Существуют также DNS-серверы, которые отображают символьные имена хостов на их IP-адреса, и без этой вспомогательной функции нормальная работа в интернете практически невозможна.

Большинство *вспомогательных (служебных)* протоколов формально относятся к прикладному уровню модели OSI, так как для передачи своих сообщений они обращаются к протоколам нижних уровней, таким как TCP, UDP или Ethernet. При этом вспомогательные протоколы переносят в сообщениях не пользовательские данные, а данные, которые необходимы для эффективной работы транспортных узлов, например для автоматического построения таблиц маршрутизации.

Очевидно, что при рассмотрении вспомогательных протоколов мы сталкиваемся с ситуацией, когда деления протоколов на уровни иерархии (то есть деления «по вертикали»), присущего модели OSI, оказывается недостаточно. Полезным оказывается деление протоколов на группы «по горизонтали».

При горизонтальном делении все протоколы верхнего уровня (как основные, так и вспомогательные) разделяют на три слоя (planes) в зависимости от того, данные какого типа они переносят (рис. 4.9):

- **пользовательский слой** (user plane) включает группу протоколов, которые переносят пользовательские данные;
- **слой управления** (control plane) составляют вспомогательные протоколы, переносящие данные, необходимые для обеспечения соединений между узлами сети; к этому слою относятся, например, протоколы маршрутизации, протоколы установления виртуальных соединений, а также протоколы отображения имен на IP-адреса;
- **слой менеджмента** (management plane) объединяет вспомогательные протоколы, переносящие данные приложений менеджмента сетью (управления сетью администратором), такие как, например, протокол SNMP для сбора информации о состоянии транспортных узлов, протокол удаленного конфигурирования устройств telnet.



Рис. 4.9. Три группы протоколов

На рис. 4.9 показано, что каждый из трех слоев имеет собственные, отличающиеся от слоя к слою, протоколы трех верхних уровней.

«Горизонтальное» деление протоколов снимает до некоторой степени сложности, возникающие при соотнесении реальных протоколов абстрактным уровням модели OSI. Например, в книгах одних авторов протоколы маршрутизации могут находиться на сетевом уровне,

в книгах других — на прикладном. Это происходит не из-за небрежности авторов, а из-за объективных трудностей классификации. Давайте вспомним, чем характеризуется идеальная многоуровневая декомпозиция. С одной стороны, необходимо соблюсти принцип иерархии: каждый вышележащий уровень обращается с запросами только к нижележащему, а нижележащий предоставляет свои сервисы только непосредственно соседствующему с ним вышележащему. В стеках протоколов это приводит к тому, что PDU вышележащего уровня всегда инкапсулируется в PDU нижележащего. С другой стороны, идеальная многоуровневая декомпозиция предполагает, что все модули, отнесенные к одному уровню, ответственны за решение общей для всех них задачи. Однако эти требования часто вступают в противоречие. Например, основной функцией протоколов сетевого уровня стека TCP/IP (так же, как и сетевого уровня OSI) является передача пакетов через составную сеть. Решение этой задачи возлагается на протокол продвижения IP-пакетов и протоколы маршрутизации, в частности RIP и OSPF. Если считать признаком принадлежности к одному и тому же уровню общность решаемых задач, то, очевидно, протокол IP и протоколы маршрутизации должны быть отнесены к одному уровню. Но если принять во внимание, что сообщения протокола маршрутизации RIP инкапсулируются в дейтаграммы транспортного уровня, а сообщения протокола OSPF — в IP-пакеты, то, формально следуя принципу иерархической организации стека, OSPF следовало бы отнести к транспортному, а RIP — к прикладному уровню.

Таким образом, модель OSI хорошо подходит для позиционирования протоколов, которые переносят пользовательский трафик, то есть протоколов пользовательского слоя, но в гораздо меньшей степени годится для определения места вспомогательных протоколов в общей модели функционирования сети.

Модель OSI и сети с коммутацией каналов

Как уже было упомянуто, модель OSI описывает процесс взаимодействия устройств в сети с *коммутацией пакетов*. А как же обстоит дело с сетями *коммутации каналов*? Существует ли для них собственная справочная модель? Можно ли сопоставить функции технологий коммутации каналов с уровнями модели OSI?

Да, для представления структуры средств межсетевого взаимодействия сетей с коммутацией каналов также используется многоуровневый подход, в соответствии с которым существуют протоколы нескольких уровней, образующих иерархию. Однако общей справочной модели, подобной модели OSI, для сетей с коммутацией каналов не существует. Например, различные типы телефонных сетей имеют собственные стеки протоколов, отличающиеся количеством уровней и распределением функций между уровнями. Первичные сети, такие как SDH или DWDM, также обладают собственной иерархией протоколов. Ситуация усложняется еще и тем, что практически все типы современных сетей с коммутацией каналов задействуют эту технику только для передачи пользовательских данных, а для управления процессом установления соединений в сети и общего управления сетью применяют технику коммутации пакетов. Такими сетями являются, например, сети ISDN, SDH, DWDM.

> Для сетей с коммутацией пакетов сети с коммутацией каналов предоставляют сервис физического уровня, хотя сами они устроены достаточно сложно и поддерживают собственную иерархию протоколов.

Рассмотрим, к примеру, случай, когда несколько локальных пакетных сетей связываются между собой через цифровую телефонную сеть. Очевидно, что функции создания составной сети выполняют протоколы сетевого уровня, поэтому мы устанавливаем в каждой локальной сети маршрутизатор. Маршрутизатор должен быть оснащен интерфейсом, способным установить соединение через телефонную сеть с другой локальной сетью. После установки соединения в телефонной сети образуется поток битов, передаваемых с постоянной скоростью. Это соединение и предоставляет маршрутизаторам сервис физического уровня. Чтобы организовать передачу данных, маршрутизаторы используют поверх этого физического канала какой-либо двухточечный протокол канального уровня.

Стандартизация сетей

Универсальный тезис о пользе стандартизации, справедливый для всех отраслей, в компьютерных сетях приобретает особое значение. Суть сети — это соединение разного оборудования, а значит, проблема совместимости является здесь одной из наиболее острых. Без согласования всеми производителями общепринятых стандартов для оборудования и протоколов прогресс в деле «строительства» сетей был бы невозможен. Поэтому все развитие компьютерной отрасли, в конечном счете, отражено в стандартах — любая новая технология только тогда приобретает «законный» статус, когда ее содержание закрепляется в соответствующем стандарте.

В компьютерных сетях идеологической основой стандартизации является рассмотренная выше модель взаимодействия открытых систем (OSI).

Понятие открытой системы

Что же такое открытая система?

Открытой может быть названа любая система (компьютер, вычислительная сеть, ОС, программный пакет, другие аппаратные и программные продукты), которая построена в соответствии с открытыми спецификациями.

Напомним, что под термином «спецификация» в вычислительной технике понимают формализованное описание аппаратных или программных компонентов, способов их функционирования, взаимодействия с другими компонентами, условий эксплуатации, особых характеристик. Понятно, что не всякая спецификация является стандартом.

> Под **открытыми спецификациями** понимаются опубликованные, общедоступные спецификации, соответствующие стандартам и принятые в результате достижения согласия после всестороннего обсуждения всеми заинтересованными сторонами.

Использование при разработке систем открытых спецификаций позволяет третьим сторонам создавать для этих систем различные аппаратные или программные средства расширения и модификации, а также программно-аппаратные комплексы из продуктов разных производителей. Большинство стандартов, принимаемых сегодня, носят открытый характер, но фирменные спецификации, точные детали которых известны только фирме-производителю, тоже сохранились. Для реальных систем полная открытость является недостижимым идеалом. Как правило, даже в системах, называемых открытыми,

этому определению соответствуют лишь некоторые части, поддерживающие внешние интерфейсы.

Модель OSI касается только одного аспекта открытости, а именно открытости средств взаимодействия устройств, связанных в компьютерную сеть. Здесь под открытой системой понимается сетевое устройство, готовое взаимодействовать с другими сетевыми устройствами по стандартным правилам, определяющим формат, содержание и значение принимаемых и отправляемых сообщений.

Если две сети построены с соблюдением принципов открытости, то это дает следующие преимущества:

- возможность построения сети из аппаратных и программных средств различных производителей, придерживающихся одного и того же стандарта;
- безболезненная замена отдельных компонентов сети другими, более совершенными, что позволяет сети развиваться с минимальными затратами;
- легкость сопряжения одной сети с другой.

Источники стандартов

Как правило, стандарты разрабатываются рабочими группами (техническими комитетами), в состав которых на добровольной основе могут включаться представители органов исполнительной власти, научных, коммерческих и некоммерческих организаций, общественных объединений. Одним из основных принципов стандартизации является ориентация на тех лиц, кто в наибольшей степени заинтересован в существовании стандартов. Поэтому очень часто разработчиками стандартов являются компании и организации, много и успешно работающие в той области, для которой они предлагают стандарты.

В зависимости от статуса организаций различают следующие виды стандартов:

- *стандарты отдельных фирм*, например стек протоколов спутниковой системы Starlink;
- *стандарты специальных комитетов и объединений* — создаются несколькими компаниями, например, стандарт беспроводных сетей Wi-Fi разработан организацией IEEE;
- *национальные стандарты*, например, с 1999 года модель OSI стала национальным стандартом РФ — ГОСТ Р ИСО/МЭК 7498-1-99 «Информационная технология. Открытые системы. Базовая эталонная модель. Спецификация»;
- *международные стандарты*, например, модель и стек коммуникационных протоколов Международной организации по стандартизации (ISO/OSI), многочисленные стандарты Международного союза электросвязи (ITU).

Некоторые стандарты, непрерывно развиваясь, могут переходить из одной категории в другую. В частности, фирменные стандарты на продукцию, получившую широкое распространение, обычно становятся международными стандартами де-факто, так как вынуждают производителей из разных стран следовать фирменным стандартам, чтобы обеспечить совместимость своих изделий с этими популярными продуктами. Например, протокол транспортного уровня QUIC, разработанный компанией Google в 2013 году, был принят в качестве стандарта интернета в 2021 году.

Более того, ввиду широкого распространения некоторые фирменные стандарты становятся основой для национальных и международных стандартов де-юре. Например, стандарт

Ethernet, первоначально разработанный компаниями Digital Equipment, Intel и Xerox, через некоторое время и в несколько измененном виде был принят как национальный стандарт IEEE 802.3, а затем организация ISO утвердила его в качестве международного стандарта ISO 8802.3.

Стандартизация интернета

Ярким примером открытой системы является интернет. Эта международная сеть развивалась в полном соответствии с требованиями, предъявляемыми к открытым системам. В разработке ее стандартов принимали и продолжают принимать участие тысячи специалистов — пользователей этой сети из различных университетов, научных организаций и фирм — производителей вычислительной аппаратуры и программного обеспечения, работающих в разных странах. Само название стандартов, определяющих работу интернета, — **темы для обсуждения** (Request For Comments, RFC) — показывает гласный и открытый характер принимаемых стандартов. В результате интернет сумел объединить в себе разнообразное оборудование и программное обеспечение огромного числа сетей, разбросанных по всему миру.

Существует несколько организационных подразделений, отвечающих за развитие и, в частности, за стандартизацию архитектуры и протоколов интернета. Основным из них является научно-административное **сообщество интернета** (Internet Society, ISOC), объединяющее более 100 000 человек, которое занимается социальными, политическими и техническими проблемами эволюции интернета.

Под управлением ISOC работает **совет по архитектуре интернета** (Internet Architecture Board, IAB). В IAB входят две основные группы: Internet Research Task Force (IRTF) и Internet Engineering Task Force (IETF). IRTF координирует долгосрочные исследовательские проекты по протоколам TCP/IP. IETF — это инженерная группа, которая занимается решением текущих технических проблем интернета. Именно IETF определяет спецификации, которые затем становятся стандартами интернета. Процесс разработки и принятия стандарта для протокола интернета состоит из ряда обязательных этапов, или стадий, включающих непременную экспериментальную проверку.

В соответствии с принципом открытости интернета все RFC-документы, в отличие, скажем, от стандартов ISO, находятся в свободном доступе. Список RFC-документов можно найти, в частности, на сайте www.rfc-editor.org.

Стандартные стеки коммуникационных протоколов

Важнейшим направлением стандартизации в области вычислительных сетей является стандартизация коммуникационных протоколов. На ранних этапах развития компьютерных сетей существовало много различных стеков коммуникационных протоколов: TCP/IP, OSI, IPX/SPX, NetBIOS/SMB, DECnet, SNA.

Разработчики модели OSI предприняли попытку создать универсальный стек протоколов, который бы полностью соответствовал их модели, — и такой стек был создан под названием стек OSI. Но этот стек не выдержал конкуренции со стеком TCP/IP, и сегодня в широком употреблении остался только один протокол стека OSI — протокол маршрутизации IS-IS (Intermediate System — Intermediate System).

Стек TCP/IP был разработан по инициативе Министерства обороны США более 45 лет назад для связи экспериментальной сети ARPAnet с другими сетями как набор общих протоколов для разнородной вычислительной среды. Большой вклад в развитие стека TCP/IP, который получил свое название по популярным протоколам сетевого уровня IP и транспортного уровня TCP, внес Университет Беркли, реализовав протоколы стека в своей версии ОС UNIX. Популярность этой операционной системы привела к широкому распространению протоколов TCP, IP и других протоколов стека в 1980–1990-е годы.

Жизнь протоколов стека TCP/IP на этом не остановилась, она просто приобрела другие формы. Стек постоянно обогащается новыми протоколами и проникает в новые области человеческой жизни. Сегодня на этом стеке работает как магистраль интернет, так и огромное количество корпоративных и домашних сетей, подключенных к интернету, а также сети интернета вещей, от сетей «умного дома» до сетей дронов. Мы подробно рассмотрим стек протоколов TCP/IP в части IV этой книги, посвященной одноименным сетям.

Говоря о стеке TCP/IP, нужно принимать во внимание тот факт, что он определяет протоколы только верхних уровней — от сетевого уровня и выше. Протоколы нижних уровней он не регламентирует, но разработан таким образом, что способен работать поверх разнообразных технологий нижних уровней (рис. 4.10). Поэтому, когда мы говорим о нижних уровнях стека TCP/IP, мы фактически говорим об очень большом количестве разнообразных технологий, которые могут быть адаптированы для включения в состав этого стека. Это могут быть как сложные и развитые технологии, например 4G, так и компактные протокольные модули, которые включаются в разные технологии, — например, модуль доступа к разделяемой среде CSMA/CA входит в состав беспроводных технологий Wi-Fi, Bluetooth, ZigBee, Thread, Wi-SUN FAN.

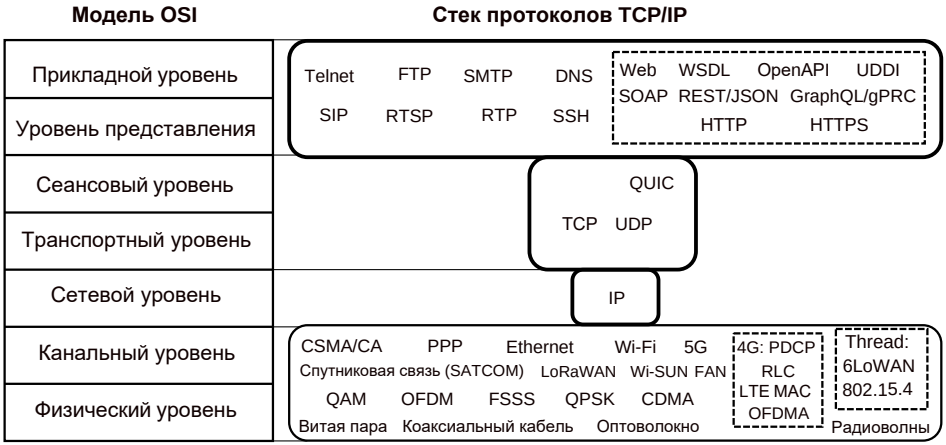


Рис. 4.10. Соответствие стека протоколов TCP/IP модели OSI

Многие из этих технологий образуют свой собственный стек протоколов, не обязательно соответствующий двум нижним уровням модели OSI. Примером такого несоответствия может служить технология сотовых сетей 4G, в которой функции физического и канального уровней реализованы четырьмя уровнями, показанными на рисунке. Стеки протоколов нижнего уровня могут образовывать иерархию стеков, примером служит стек технологии Thread, разработанной для управления устройствами умного дома интернета вещей. Этот

стек использует на нижнем уровне протоколы стека IEEE 802.15.4, а для того, чтобы вписаться в стек TCP/IP, эта технология использует протокол адаптации 6LoWAN, поверх которого может работать протокол IP.

На верхних уровнях стека TCP/IP также имеется большое разнообразие протоколов. В первую очередь это традиционные, существующие с первых дней интернета протоколы: почтовой службы SMTP, обмена файлами FTP, удаленного управления маршрутизаторами Telnet. Позже к ним прибавились протоколы интернет-телефонии SIP, RSTP и протоколы веб-сервисов. Протоколы веб-сервисов образовали свой собственный трехуровневый стек протоколов.

Как видим, за названием «стек протоколов TCP/IP» стоит множество вариантов этого стека, отличающихся как протоколами нижних уровней, так и верхних. И только *один протокол всегда присутствует в любом варианте стека TCP/IP — это протокол IP*, который служит своеобразным «клеем», позволяющим различным протоколам и стекам протоколов работать совместно. В этом смысле протокол IP занимает уникальное место в мире сетевых технологий. Внешний вид диаграммы современного состояния стека TCP/IP стал причиной того, что его иногда называют «стеком с тонкой талией» (narrow waist stack) или «стеком в форме песочных часов».

Однако существуют немногочисленные стеки, в которых сетевой уровень либо вообще отсутствует, либо на этом уровне работает другой протокол, отличный от IP. Примером стеков первого типа являются стеки технологий, в которых маршрутизация не нужна, например стек беспроводной технологии Bluetooth, в которой все устройства разделяют одну и ту же радиосреду и поэтому могут общаться друг с другом непосредственно. Технология ZigBee, созданная для автоматизации устройств умного дома, применяет свой собственный стек протоколов, в котором на сетевом уровне работает собственный протокол. Возможно, такие исключения носят временный характер, так, на смену технологии ZigBee приходит технология Thread, которая решает те же задачи, но на основе протокола IP.

Классификация компьютерных сетей

Классификация — это процесс группирования, то есть отнесения к тому или иному типу объектов изучения в соответствии с их общими признаками.

Каждый реальный объект может быть наделен множеством признаков. *Субъективный* характер любой классификации проявляется в том, что имеется некоторый произвол при выборе среди этого множества признаков тех, которые будут использованы для классификации, то есть при выборе **критериев классификации**. Приведенная далее классификация компьютерных сетей не является исключением — в других источниках (да и далее в этой книге) вы можете встретить другие признаки, по которым сети относят к тому или иному типу.

Начнем с того, что компьютерные сети сами являются результатом классификации **телекоммуникационных сетей**, которые по виду передаваемого контента делятся на:

- радиосети;
- телефонные сети;

- телевизионные сети;
- компьютерные сети.

В зависимости от *территории покрытия* компьютерные сети можно разделить на три группы:

- **локальные сети** (Local Area Network, LAN);
- **глобальные сети** (Wide Area Network, WAN);
- **городские сети**, или **сети мегаполиса** (Metropolitan Area Network, MAN).

Мы уже обозначили особенности этих групп сетей, когда рассматривали в главе 1 эволюцию компьютерных сетей. В частности, в локальных сетях качество линий связи между узлами обычно выше, чем в глобальных сетях. Это обусловлено различными причинами:

- существенно меньшей длиной линий связи (метры вместо сотен километров), а значит, и меньшими искажениями сигналов, вносимых неидеальной передающей средой;
- меньшим уровнем внешних помех, так как в локальной сети оборудование и кабели обычно размещаются в специальных защищенных экранированных помещениях, а линии связи глобальной сети могут проходить в сильно электромагнитно «зашумленной» среде, например, в туннелях подземных коммуникаций, рядом с силовыми кабелями, вдоль линий электропередач и т. п.;
- экономическими соображениями.

Высокое качество линий связи и низкий уровень помех позволили упростить процедуры передачи данных в технологиях локальных сетей, например, применять простые методы кодирования и модулирования сигналов, отказаться от сложных алгоритмов восстановления искаженных данных.

Несмотря на то что процесс сближения технологий локальных и глобальных сетей идет уже давно, различия между этими технологиями все еще достаточно отчетливы, что и дает основания относить соответствующие сети к различным технологическим типам.

Сети MAN предназначены для обслуживания территории крупного города — мегаполиса и сочетают в себе признаки как локальных, так и глобальных сетей. От первых они унаследовали большую плотность подключения конечных абонентов и высокоскоростные линии связи, а от последних — большую протяженность линий связи.

В соответствии с технологическими признаками, обусловленными *средой передачи*, компьютерные сети подразделяют на два класса:

- **проводные сети** — сети, каналы связи которых построены с использованием медных или оптических кабелей;
- **беспроводные сети** — сети, в которых для связи используются беспроводные каналы связи, например радио, СВЧ, инфракрасные или лазерные каналы.

Любая беспроводная среда — будь то радиоволны, инфракрасные лучи или СВЧ-сигналы спутниковой связи — гораздо больше подвержена влиянию внешних помех, чем проводная. Роса, туман, солнечные бури, работающие в комнате микроволновые печи — вот только несколько примеров источников помех, которые могут привести к резкому ухудшению качества беспроводного канала. А значит, технологии беспроводных сетей должны учитывать типичность подобных ситуаций и строиться таким образом, чтобы обеспечивать

работоспособность сети, несмотря на ухудшение внешних условий. Здесь, как и в случае локальных и глобальных сетей, мы сталкиваемся с влиянием качества линий связи на технологию передачи данных.

Кроме того, существует ряд других специфических особенностей беспроводных сетей, которые служат основанием для выделения их в особый класс, например, естественное разделение радиосреды всеми узлами сети, находящимися в радиусе действия всенаправленного передатчика; распределение диапазона радиочастот между сетями различного назначения, например, между телефонными и компьютерными.

Огромное разнообразие беспроводных сетей — спутниковые и сотовые сети, сети доступа Wi-Fi и мобильные AdHoc-сети MANET, сети управления дронами и интернет вещей — все это приводит к необходимости отдельной классификации сетей этого класса, которую можно найти в части VI «Беспроводная передача данных».

В зависимости от способа *коммутации*, сети подразделяются на два фундаментально различных класса:

- **сети с коммутацией пакетов**;
- **сети с коммутацией каналов**.

Сейчас в компьютерных сетях используется преимущественно техника коммутации пакетов, хотя принципиально допустимо и применение в них техники коммутации каналов.

В свою очередь, техника коммутации пакетов допускает несколько вариаций, отличающихся способом продвижения пакетов, в соответствии с чем сети делятся на:

- **дейтаграммные сети**, например Ethernet;
- **сети, основанные на логических соединениях**, например IP-сети, использующие на транспортном уровне протокол TCP;
- **сети, основанные на виртуальных каналах**, например MPLS-сети.

Сети могут быть классифицированы на основе *топологии*. Топологический тип сети весьма отчетливо характеризует сеть, он понятен как профессионалам, так и пользователям. Мы подробно рассматривали базовые топологии сетей, поэтому здесь только перечислим их: **полносвязная топология**, **дерево**, **звезда**, **кольцо**, **смешанная топология**.

В зависимости от того, *какому типу пользователей предназначаются услуги сети*, последние делятся на сети операторов связи, корпоративные и персональные сети.

- **Сети операторов связи** предоставляют публичные услуги, то есть клиентом сети может стать любой индивидуальный пользователь или любая организация, которая заключила соответствующий коммерческий договор на предоставление той или иной телекоммуникационной услуги. Традиционными услугами операторов связи являются услуги телефонии, а также предоставления каналов связи в аренду тем организациям, которые собираются строить на их основе собственные сети. С распространением компьютерных сетей операторы связи существенно расширили спектр своих услуг, добавив к ним услуги интернета, услуги виртуальных частных сетей, веб-хостинг, электронную почту и IP-телефонию, а также широковещательную рассылку аудио- и видеосигналов. Сегодня интернет также является одной из разновидностей сети операторов связи. Интернет быстро превратился из сети, обслуживающей сравнительно немногочисленное академическое сообщество, во всемирную публичную сеть, предоставляющую набор наиболее востребованных услуг для всех.

- **Корпоративные сети** предоставляют услуги только сотрудникам предприятия, которое владеет этой сетью. Хотя формально корпоративная сеть может иметь любой размер, обычно под корпоративной понимают сеть крупного предприятия, которая состоит как из локальных сетей, так и из объединяющей их глобальной сети.
- **Персональные сети** находятся в личном использовании. Для них характерно небольшое количество узлов, простая структура, а также небольшой (в пределах 30 метров) радиус действия. Узлами персональной сети наряду с настольными компьютерами могут быть телефоны, смартфоны, планшеты, ноутбуки. Чаще всего персональные сети строятся на основе беспроводных технологий.

В зависимости от *функциональной роли*, которую играют некоторые части сети, ее относят к сети доступа, магистральной сети или сети агрегирования трафика (рис. 4.11). Поясним, что:

- **Сети доступа** — это сети, предоставляющие доступ индивидуальным и корпоративным абонентам от их помещений (квартир, офисов) до первого помещения (пункта присутствия) оператора сети связи или оператора корпоративной сети. Другими словами, это сети, ответственные за расширение глобальной сети до помещений ее клиентов.
- **Магистральные сети** — это сети, представляющие собой наиболее скоростную часть (ядро) глобальной сети, которая объединяет многочисленные сети доступа в единую сеть.

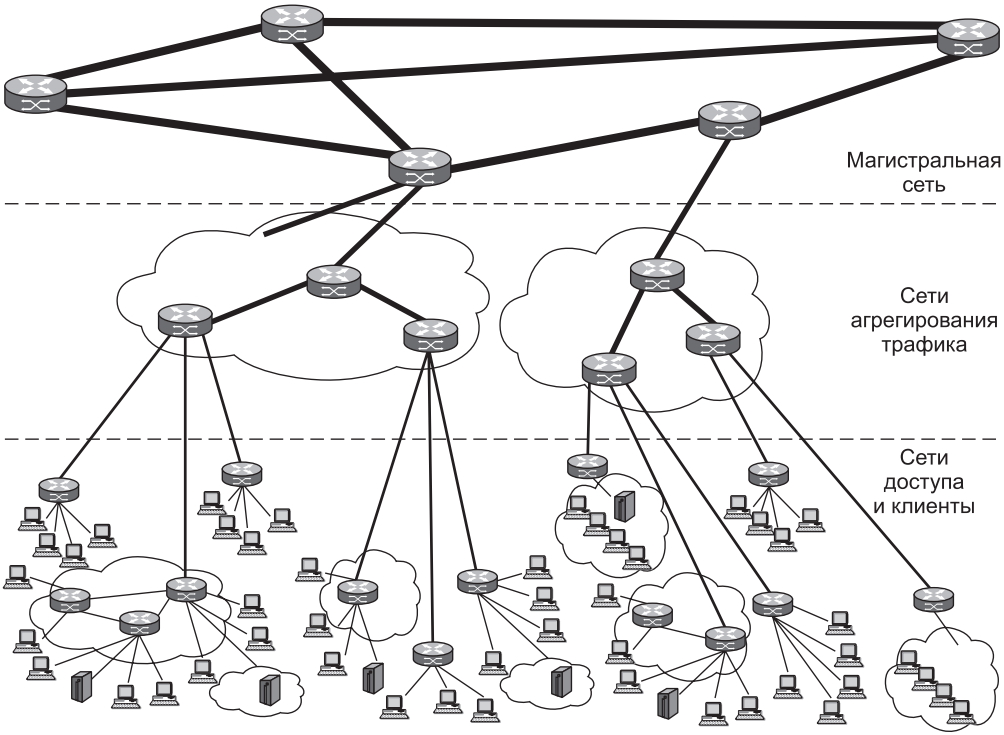


Рис. 4.11. Сети доступа, сети агрегирования трафика и магистральная сеть

- **Сети агрегирования трафика** — это сети, агрегирующие данные от многочисленных сетей доступа для компактной передачи их по небольшому числу каналов связи в магистраль. Сети агрегирования обычно используются только в крупных глобальных сетях, где они занимают промежуточную позицию, помогая магистральной сети обрабатывать трафик, поступающий от большого числа сетей доступа. В сетях среднего и небольшого размера сети агрегирования обычно отсутствуют.

Различают также первичные и наложенные телекоммуникационные сети:

- **Первичные сети** занимают особое положение в мире телекоммуникационных сетей, их можно рассматривать как вспомогательные сети, позволяющие гибко создавать постоянные физические двухточечные каналы для других компьютерных и телефонных сетей. В соответствии с семиуровневой моделью OSI первичные сети, подобно простым кабелям, выполняют функции физического уровня сетей. Однако в отличие от кабелей первичные сети включают дополнительное коммуникационное оборудование, которое путем соответствующего конфигурирования позволяет прокладывать новые физические каналы между конечными точками сети. Другими словами, первичная сеть — это *гибкая среда для создания физических каналов связи.*
- **Наложенные сети** в этой классификации — это все остальные сети, которые предоставляют услуги конечным пользователям и строятся на основе каналов первичных сетей — «накладываются» поверх этих сетей. Так, и компьютерные, и телефонные, и телевизионные сети являются наложенными.

Оптоволоконные кабели обладают наилучшими на сегодняшний день характеристиками передачи данных, они используются как в локальных, так и в глобальных проводных сетях. Тем не менее термин **оптические сети** часто трактуется специалистами по компьютерным сетям и телекоммуникациям в узком смысле как синоним первичных сетей. Это объясняется тем, что передача данных по оптическим кабелям является для первичных сетей основным вариантом работы, а использование других сред передачи не может обеспечить в первичных сетях высоких современных требований по скорости и надежности обмена информацией между удаленными узлами сети.

Интернет представляет собой уникальную сеть, объединяющую практически все компьютерные сети (за исключением, может быть, сетей, остающихся изолированными по причине повышенной секретности) во всемирном масштабе. Если применить к интернету признаки, описанные в классификации, можно сказать, что это:

- сеть операторов связи, предоставляющая публичные услуги — как информационные, так и транспортные;
- сеть с коммутацией пакетов;
- сеть, состоящая из магистральных сетей, сетей агрегирования трафика и сетей доступа.

ГЛАВА 5 Сетевые характеристики и качество обслуживания

Компьютерная сеть представляет собой сложную и дорогую систему, решающую ответственные задачи и обслуживающую большое количество пользователей. Поэтому очень важно, чтобы сеть не просто работала, а работала качественно.

Понятие *качества обслуживания* можно трактовать очень широко, включая в него все возможные и желательные для пользователя свойства сети и поставщика услуг, поддерживающего работу этой сети. Чтобы пользователь и поставщик услуг могли более конкретно обсуждать проблемы обслуживания и строить свои отношения на формальной основе, существует ряд общепринятых характеристик качества предоставляемых сетью услуг. В этой главе мы рассмотрим только характеристики качества *транспортных* услуг сети, которые намного проще поддаются формализации, чем характеристики качества информационных услуг. Характеристики качества транспортных услуг отражают такие важнейшие свойства сети, как производительность, надежность и безопасность.

Часть этих характеристик может быть оценена количественно и измерена при обслуживании пользователя. Пользователь и поставщик услуг могут заключить соглашение об уровне обслуживания, определяющем требования к количественным значениям некоторых характеристик, например, к доступности предоставляемых услуг.

Типы характеристик

Субъективные оценки качества

Если опросить пользователей о том, что они вкладывают в понятие качественных сетевых услуг, то можно получить очень широкий спектр ответов. Среди них, скорее всего, встретятся следующие мнения:

- сеть работает быстро, без задержек;
- трафик передается надежно, данные не теряются;
- услуги предоставляются бесперебойно по схеме 24 × 7 (то есть 24 часа в сутки семь дней в неделю);
- служба поддержки работает хорошо, давая полезные советы и помогая разрешить проблемы;
- услуги предоставляются по гибкой схеме, мне нравится, что можно в любой момент и в широких пределах повысить скорость доступа к сети и увеличить число точек доступа;
- поставщик не только передает мой трафик, но и защищает мою сеть от вирусов и атак злоумышленников;

- я всегда могу проконтролировать, насколько быстро и без потерь сеть передает мой трафик;
- поставщик предоставляет широкий спектр услуг, в частности, помимо стандартного доступа в интернет он предлагает хостинг для моего персонального веб-сайта и услуги IP-телефонии.

Эти *субъективные* оценки отражают *пожелания пользователей* к качеству сетевых сервисов.

Требования к характеристикам со стороны пользователя и поставщика услуг

Пользователи сети — это хотя и важная, но только одна сторона бизнеса сетей передачи данных. Другая сторона — поставщик услуг: коммерческий, если это публичная сеть, или некоммерческий, если сеть корпоративная. Чтобы обе стороны — пользователи и поставщики услуг — могли «найти общий язык», существуют *формализованные количественные характеристики* качества сетевых услуг.

Получая сетевые услуги, пользователь формулирует определенные *требования к характеристикам сети*. Например, пользователь может потребовать, чтобы средняя скорость передачи его информации через сеть не опускалась ниже 2 Мбит/с. То есть в данном случае пользователь задает тот диапазон значений для средней скорости передачи информации через сеть, который для него означает хорошее качество сервиса.

Все множество количественных характеристик качества транспортных услуг сети можно свести к желанию пользователя без потерь и перерывов в обслуживании **(надежность)** передавать с заданной скоростью **(производительность)** защищенную от несанкционированного доступа и подмены **(безопасность)** информацию.

Понятно, что поставщик сетевых услуг, стремясь удовлетворить требования клиентов, также уделяет внимание этим характеристикам. В то же время существует ряд характеристик, важных для поставщика сети, но не представляющих интереса для пользователей. Поскольку сеть обслуживает большое количество клиентов, информационные потоки которых разделяют ресурсы сети — линии связи и коммутаторы (маршрутизаторы), — поставщику необходимо найти такой баланс в распределении ресурсов между конкурирующими потоками, чтобы требования *всех* пользователей были соблюдены. Решение этой задачи включает планирование и контроль расходования ресурсов в процессе передачи пользовательского трафика. В связи с этим для поставщика к числу важных характеристик относится производительность оборудования сети — например, производительность коммутатора дает ему возможность оценить, какое количество потоков пользователей способен обработать этот коммутатор. Для пользователя же производительность коммутатора никакого значения не имеет, ему важен конечный результат — будет его поток обслужен качественно или нет.

Долговременные, среднесрочные и краткосрочные характеристики

Рассмотрим еще один способ классификации характеристик — в соответствии с временно́й шкалой, на которой эти характеристики определяются.

Долговременные характеристики (или **характеристики проектных решений**) определяются на промежутках времени от нескольких месяцев до нескольких лет. Примерами таких характеристик являются количество и схема соединения коммутаторов в сети, пропускная способность линий связи, конкретные модели и характеристики используемого оборудования. Эти параметры сети прямо влияют на характеристики качества сетевых услуг. Понятно, что полная замена или глубокая модернизация сети связаны с большими финансовыми и временны́ми затратами, поэтому они не могут происходить часто, а значит, выбранные однажды параметры продолжают оказывать влияние на качество функционирования сети в течение продолжительного времени.

Среднесрочные характеристики определяются на интервалах времени от нескольких секунд до нескольких дней. Как правило, за это время происходит обслуживание большого количества пакетов. Например, к среднесрочным характеристикам может быть отнесено усредненное значение задержки пакетов по выборке, взятой в течение суток.

Краткосрочные характеристики относятся к темпу обработки отдельных пакетов и измеряются в микросекундном и миллисекундном диапазонах. Например, время буферизации или время пребывания пакета в очереди коммутатора либо маршрутизатора является характеристикой этой группы. Для анализа и обеспечения требуемого уровня краткосрочных характеристик разработано большое количество методов, получивших название методов **контроля и предотвращения перегрузок**.

Соглашение об уровне обслуживания

Основой нормального сотрудничества поставщика услуг и пользователей является *договор*. Он заключается всегда, однако далеко не всегда в нем указываются количественные требования к эффективности предоставляемых услуг. Очень часто в договоре услуга определяется, например, как «предоставление доступа в интернет» и т. п.

Однако существует и другой тип договора, называемый **соглашением об уровне обслуживания** (Service Level Agreement, **SLA**). В таком соглашении поставщик услуг и клиент описывают качество предоставляемой услуги в количественных терминах, пользуясь характеристиками эффективности сети. Например, в SLA может быть записано, что поставщик обязан передавать трафик клиента без потерь и с той средней скоростью, с которой пользователь направляет его в сеть. При этом оговорено, что это соглашение действует только в том случае, если средняя скорость трафика пользователя не превышает, например, 3 Мбит/с, в противном случае поставщик получает право просто не передавать избыточный трафик. Чтобы каждая сторона могла контролировать соблюдение этого соглашения, необходимо указать и период времени, в течение которого будет измеряться средняя скорость, например день, час или секунда. Еще более определенным соглашение об уровне обслуживания становится в том случае, когда в нем указываются средства и методы измерения характеристик сети — с тем, чтобы у поставщика и пользователя не было расхождений при контроле соглашения.

Соглашения об уровне обслуживания могут заключаться не только между поставщиками коммерческих услуг и их клиентами, но и между подразделениями одного и того же предприятия. В этом случае поставщиком сетевых услуг может являться, например, отдел информационных технологий, а потребителем — производственный отдел.

Производительность и надежность сети

Идеальная и реальная сети

В главе 3 (раздел «Количественное сравнение задержек. Пример») мы рассмотрели различные составляющие *задержек* в сети с коммутацией пакетов, а именно:

- время передачи данных в канал (время сериализации);
- время распространения сигнала;
- время ожидания пакета в очереди;
- время коммутации пакета.

Две первые составляющие задержки полностью определяются пропускной способностью каналов передачи данных и являются *фиксированными* величинами для пакета фиксированной длины. Две последние составляющие зависят от загрузки коммутаторов и их быстродействия и для пакета фиксированной длины в общем случае являются *переменными*.

Будем считать, что сеть с коммутацией пакетов работает идеально, если она передает каждый бит информации с постоянной скоростью, равной скорости распространения света в используемой физической среде. Другими словами, *идеальная сеть с коммутацией пакетов не вносит никаких дополнительных задержек в передачу данных помимо тех, которые вносятся каналами связи*, то есть две последние составляющие задержки равны нулю.

Результат передачи пакетов такой идеальной сетью иллюстрирует рис. 5.1. На верхней оси показаны значения времени поступления пакетов в сеть от узла отправителя, а на нижней — значения времени поступления пакетов в узел назначения. Таким образом, верхняя ось показывает **предложенную нагрузку** сети, а нижняя — результат передачи этой нагрузки через сеть.

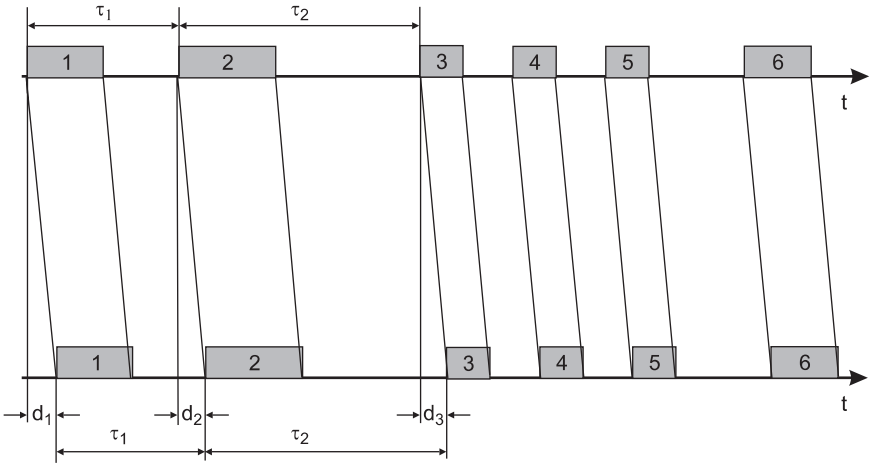


Рис. 5.1. Передача пакетов идеальной сетью

Определим **время задержки пакета** как интервал времени между моментом отправления *первого* бита пакета в канал связи узлом отправления и моментом поступления *первого*

бита пакета в узел назначения соответственно (на рисунке обозначены задержки d_1, d_2 и d_3 пакетов 1, 2 и 3 соответственно), то есть задержка пакета равна времени распространения сигнала.

Как видно из рисунка, идеальная сеть доставляет все пакеты узлу назначения:

- не потеряв ни одного из них (и не исказив информацию ни в одном из них);
- в том порядке, в котором они были отправлены;
- с одной и той же минимально возможной задержкой ($d_1 = d_2$ и т. д.).

Важно, что все интервалы между соседними пакетами сеть сохраняет в неизменном виде. Например, если интервал между первым и вторым пакетами составляет при отправлении τ_1 секунд, а между вторым и третьим — τ_2, то такими же интервалы останутся и в узле назначения.

Надежная доставка всех пакетов с минимально возможной задержкой и сохранением временны́х интервалов между ними удовлетворит любого пользователя сети независимо от того, трафик какого приложения он передает по сети — веб-сервиса или IP-телефонии.

ПРИМЕЧАНИЕ

Существуют и другие определения времени задержки пакета. Например, эту величину можно определить как время между моментом отправления первого бита пакета в канал связи узлом отправления и моментом поступления последнего бита пакета в узел назначения соответственно (такое определение используется в документе RFC 2679, описывающем характеристики задержек IP-пакетов). Нетрудно заметить, что в этом определении в задержку пакета включено время сериализации.

Теперь посмотрим, какие отклонения от идеала могут встречаться в *реальной* сети и какими характеристиками можно эти отклонения описывать (рис. 5.2).

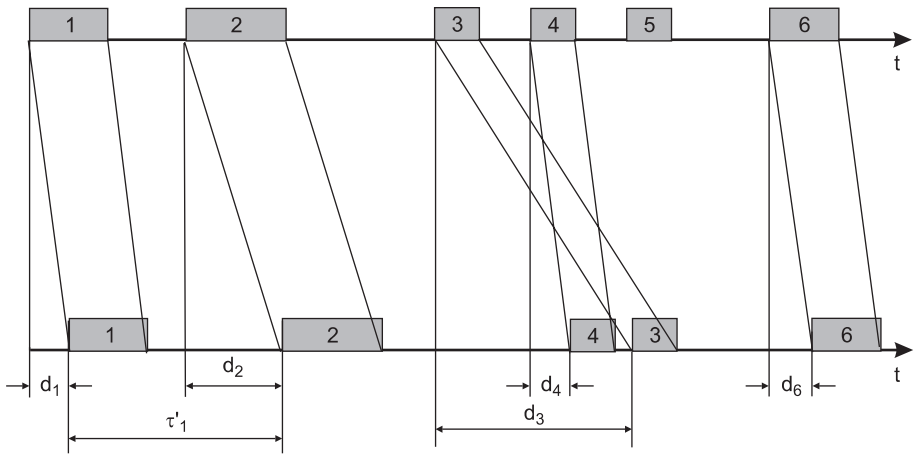


Рис. 5.2. Передача пакетов реальной сетью

Пакеты доставляются сетью узлу назначения *с различными* задержками. Это — неотъемлемое свойство сетей с коммутацией пакетов.

Случайный характер процесса образования очередей приводит к *случайным* задержкам, при этом задержки отдельных пакетов могут быть значительными, в десятки раз превосходя среднюю величину задержек ($d_1 \neq d_2 \neq d_3$ и т. д.). Неравномерность задержек изменяет относительное положение пакетов в выходном потоке, что может катастрофически сказаться на качестве работы некоторых приложений. Например, при цифровой передаче речи исходный поток представляет собой равномерно отстоящие друг от друга пакеты, несущие замеры голоса. Неравномерность интервалов между пакетами выходного потока приводит к существенным искажениям речи.

Пакеты могут доставляться узлу назначения *не в том порядке*, в котором они были отправлены, — например, на рис. 5.2 пакет 4 поступил в узел назначения раньше, чем пакет 3. Такие ситуации встречаются в дейтаграммных сетях, когда различные пакеты одного потока передаются через сеть различными маршрутами и, следовательно, ожидают обслуживания в разных очередях с разным уровнем задержек. Очевидно, что пакет 3 проходил через перегруженный узел или узлы, так что его суммарная задержка оказалась настолько большой, что пакет 4 прибыл раньше него.

Пакеты *могут теряться* в сети или же приходить в узел назначения *с искаженными данными*, что равносильно потере пакета, так как большинство протоколов неспособно восстанавливать искаженные данные, лишь определяя этот факт по значению контрольной суммы в заголовке кадра.

Пакеты также могут *дублироваться* по разным причинам, например, из-за ошибочных повторных передач пакета, предпринятых протоколом, в котором таким образом обеспечивается надежный обмен данными.

В реальной сети скорость информационного потока на входе узла назначения может отличаться от скорости потока, направленного в сеть узлом-отправителем. Виной этому являются задержки пакетов в очередях и их потери. Так, в примере, показанном на рис. 5.2, *средняя скорость исходящего потока снижается* из-за потери пакета 5. Чем больше потерь и искажений пакетов происходит в сети, тем ниже скорость информационного потока.

Для оценки отклонения функционирования реальной сети от идеальной используются различные **характеристики производительности сети**. Одни из них касаются различных аспектов скорости и задержек, другие — потерь и искажений, третьи — безопасности передачи информации. Относительная важность той или иной характеристики зависит от типа приложения, трафик которого переносит сеть. Так, существуют приложения, которые очень чувствительны к задержкам пакетов, но в то же время весьма терпимы к потере отдельного пакета — примером может служить передача голоса через пакетную сеть. Поэтому для каждого конкретного случая необходимо выбирать подходящий набор характеристик сети, адекватно отражающий влияние «неидеальности» сети на работу приложения.

Статистические оценки характеристик сети

Для оценки характеристик *случайных процессов* служат статистические методы, а именно такой характер имеют процессы передачи пакетов сетью. Сами характеристики производительности сети — как, например, задержка пакета — являются *случайными величинами*. Статистические характеристики выявляют закономерности в поведении сети, которые устойчиво проявляются только на длительных периодах времени. Когда мы говорим о длительном периоде времени, то мы понимаем под этим интервал, в миллионы раз больший,

чем время передачи одного пакета, которое в современной сети измеряется микросекундами, например, время передачи пакета Gigabit Ethernet составляет около 10 мкс. Поэтому для получения устойчивых результатов нужно наблюдать поведение сети, по крайней мере, в течение нескольких минут, а лучше — нескольких часов.

Основным инструментом статистики является так называемая **гистограмма** распределения оцениваемой случайной величины. Рассмотрим, например, гистограмму задержки пакета. Будем считать, что нам удалось измерить задержку доставки каждого из 2600 пакетов, переданных между двумя узлами сети, и сохранить полученные результаты. Каждый результат измерения является *единичным значением* случайной величины, а в совокупности они образуют **выборку** значений случайной величины.

Чтобы получить гистограмму распределения, мы должны разбить весь диапазон измеренных значений задержек на несколько интервалов и подсчитать, сколько замеров из выборки попало в каждый интервал. Пусть все значения задержек укладываются в диапазон 20–90 мс. Разобьем его на семь интервалов по 10 мс. В каждый из этих интервалов, начиная с интервала 20–30 мс и т. д., попало 100 ($n1$), 200 ($n2$), 300 ($n3$), 300 ($n4$), 400 ($n5$), 800 ($n6$) и 500 ($n7$) пакетов соответственно. Отобразив эти числа в виде горизонтальных уровней для каждого интервала, мы получим гистограмму, показанную на рис. 5.3, которая, основываясь всего на семи числах $n1$, $n2$, ... $n7$, дает нам компактную статистическую характеристику задержек 2600 пакетов.

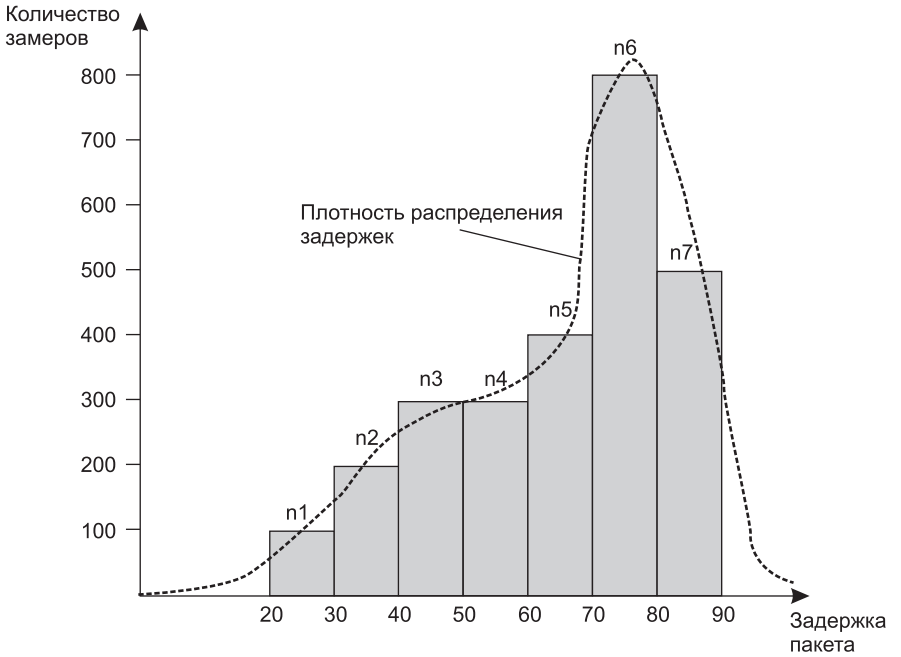


Рис. 5.3. Гистограмма распределения задержек

Гистограмма задержек дает хорошее представление о производительности сети. По ней можно судить, какие уровни задержек более вероятны, а какие — менее. Чем больше период времени, в течение которого собираются данные для построения гистограммы, тем с более

высокой степенью достоверности можно предсказать поведение сети в будущем. Например, пользуясь гистограммой на рис. 5.3, можно сказать, что и в будущем при измерениях задержек пакетов у 65 % пакетов задержка превысит 60 мс. Для получения такой оценки мы сложили общее количество пакетов, задержки которых попали во все интервалы, превышающие 60 мс (1700 замеров), и разделили эту величину на общее количество пакетов (2600 замеров).

Насколько точен такой прогноз? Собрали ли мы достаточно экспериментальных данных, чтобы делать более-менее достоверные прогнозы? Статистика позволяет судить и об этом, однако мы не будем рассматривать здесь эту увлекательную проблему (желающие могут обратиться к специальным книгам по статистике).

При увеличении количества интервалов и времени наблюдения гистограмма в пределе переходит в непрерывную функцию, которая называется **плотностью распределения** задержки доставки пакета (показана пунктиром). В соответствии с теорией вероятность того, что значение случайной величины окажется в определенном диапазоне, равна интегралу плотности распределения случайной величины от нижней до верхней границы данного диапазона. Таким образом, может быть вычислено вероятностное значение задержки пакета.

Гистограмма дает наглядное представление соответствующей характеристики, но чаще используются более компактные **статистические оценки** характеристик, которые позволяют представить характеристику *одним числом* на основе некоторой математической обработки имеющейся выборки.

Для описания характеристик сети используются следующие статистические оценки:

- **среднее значение** (D) вычисляется как сумма всех значений оцениваемой величины d_i, деленная на количество всех измерений N:

$$D = \Sigma \frac{d_i}{N}.$$

  Для примера, приведенного на рис. 5.3, среднее значение равно: (100 × 25 + 200 × 35 + + 300 × 45 + 300 × 55 + 400 × 65 + 800 × 75 + 500 × 85)/2600 = 64,6 мс (при вычислениях использованы средние значения интервалов);

- **медиана** представляет такое значение оцениваемой величины, которое делит ранжированную (упорядоченную) выборку пополам, то есть таким образом, чтобы количество замеров, значения которых меньше или равны значению медианы, равнялось количеству замеров, значения которых больше или равны значению медианы. В нашем примере медианой выборки является значение 70 мс, поскольку число замеров, значения которых меньше или равны 70 мс, составляет 1300, как и число замеров, значения которых больше или равны 70 мс;

- **стандартное отклонение, или вариация**, (J) представляет собой среднее отклонение каждого отдельного замера от среднего значения оцениваемой величины:

$$J = \sqrt{\frac{\Sigma (d_i - D)^2}{N - 1}}.$$

  Очевидно, что если все задержки d_i равны между собой, то вариация отсутствует, что подтверждают приведенные формулы — в этом случае $D = d_i$ и $J = 0$;

- **коэффициент вариации** (*CV*) — это безразмерная величина, которая равна отношению стандартного отклонения к среднему значению оцениваемой величины:

$$CV = \frac{J}{D}.$$

  Коэффициент вариации характеризует оцениваемую величину без привязки к ее абсолютным значениям. Так, идеальный равномерный поток пакетов всегда будет обладать нулевым значением коэффициента вариации задержки пакета. Коэффициент вариации задержки пакета, равный 1, означает пульсирующий трафик, так как средние отклонения интервалов от некоторого среднего периода следования пакетов равны этому периоду;

- **квантиль (процентиль)** — это значение оцениваемой величины, делящее ранжированную выборку на две части так, что процент замеров, значения которых меньше или равны значению квантиля, равен некоторому заданному уровню. В этом определении фигурируют два числа: заранее заданный процент и найденное по нему и замерам выборки значение квантиля. Нетрудно заметить, что 50-процентный квантиль равен медиане. Для оценки характеристик сети обычно используют квантили с достаточно большим значением процента, например 99-процентные квантили, которые дают возможность судить о наиболее вероятных значениях характеристик.

Статистические методы применяются к различным характеристикам производительности сети — скорости передачи данных, задержкам, времени ожидания в буфере, времени коммутации, количеству потерянных пакетов и др., так как все они являются случайными величинами.

Последовательность замеров рекомендуется выполнять в случайные моменты времени, подчиняющиеся распределению Пуассона. Такой порядок выбора времени замеров позволяет избежать возможной синхронизации измерений с любыми периодическими флуктуациями в поведении сети, так как подобная синхронизация может существенно исказить наблюдаемую картину.

Активные и пассивные измерения в сети

Чтобы оценить некоторую характеристику производительности сети, необходимо провести определенные измерения на последовательности пакетов, поступающих на некоторый интерфейс сетевого устройства. Существует два типа измерений в сети: активные и пассивные.

Активные измерения основаны на генерации в узле-источнике специальных «измерительных» пакетов. Эти пакеты должны пройти через сеть тем же путем, что и пакеты, характеристики которых мы собираемся оценивать. Измерения в узле назначения проводятся на последовательности «измерительных» пакетов.

Рисунок 5.4 иллюстрирует идею активных измерений. Допустим, требуется измерить задержки пакетов, которые передаются от компьютера-клиента приложения *A* компьютеру-серверу приложения *A* через сеть. Вместо того чтобы пытаться измерить задержки пакетов, генерируемых клиентским компьютером, мы устанавливаем в сети два дополнительных компьютера: *сервер-генератор* и *сервер-измеритель*. Сервер-генератор генерирует измерительные пакеты (показанные на рисунке серым цветом), сервер-измеритель — измеряет задержки этих пакетов.

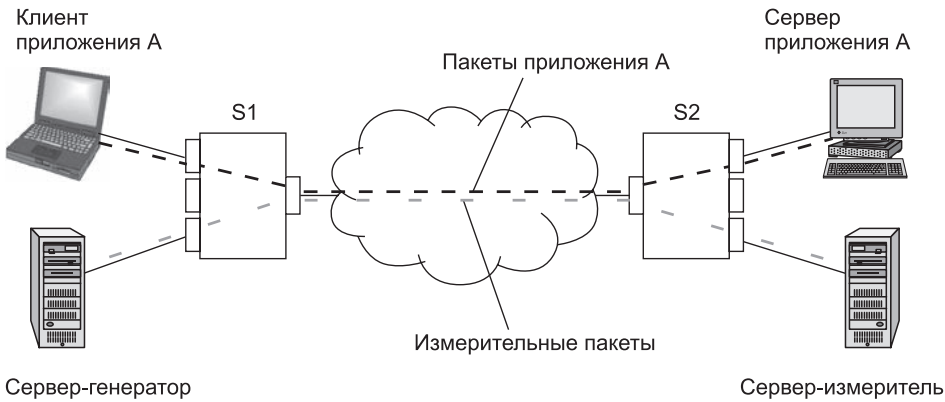


Рис. 5.4. Схема активных измерений

Чтобы измеряемые значения были близки к значениям задержки пакетов приложения *A*, нужно, чтобы измерительные пакеты проходили через сеть по тому же пути, что и пакеты приложения *A*. В нашем примере эта цель достигается за счет подключения дополнительных компьютеров к портам тех же коммутаторов *S*1 и *S*2, к которым подключены оригинальные узлы. Кроме того, требуется, чтобы измерительные пакеты как можно больше «походили» на оригинальные пакеты — размерами, признаками, помещенными в заголовки пакетов.

Измерительные пакеты не должны генерироваться слишком часто, иначе нагрузка сети может существенно измениться и результаты замеров будут отличаться от тех, которые были бы получены в отсутствие измерительных пакетов. Другими словами, измерения не должны менять условий работы сети. Обычно интенсивность генерации измерительных пакетов не превосходит 20–50 пакетов в секунду.

Возникает естественный вопрос, зачем нужно решать столько лишних проблем: размещать дополнительное оборудование, создавать условия для измерительных пакетов, близкие к условиям обработки оригинальных пакетов, и в то же время стараться не изменить нагрузку сети? Не проще ли измерять параметры реальных пакетов? Ответ заключается в том, что *активная схема упрощает процесс проведения измерений и позволяет добиться их высокой точности.*

Во-первых, так как сервер-генератор создает специальные пакеты, предназначенные для измерений, он легко может поместить в них служебную информацию, например, временну́ю отметку (time-stamp) отправки пакета, с тем чтобы сервер-измеритель использовал ее для вычисления времени задержки.

Во-вторых, для того чтобы измерения задержки были точными, нужна хорошая синхронизация сервера-генератора и сервера-измерителя. Поскольку в схеме активных измерений они представляют собой специальные узлы, такой синхронизации добиться проще, чем в случае синхронизации клиентской и серверной частей приложения *A*, которые чаще всего установлены на обычных компьютерах.

В-третьих, иногда у инженеров, проводящих измерения, просто нет доступа к компьютерам, на которых работают приложения, чтобы установить там программное обеспечение для требуемых измерений.

В-четвертых, если такой доступ и существует, то операционные системы клиента и сервера, как и их аппаратная платформа, скорее всего, не оптимизированы для точных измерений временны́х интервалов, а значит, вносят большие искажения в результаты (например, за счет задержек программы измерений в очереди к центральному процессору).

Однако преимущества активной схемы измерений не являются абсолютными. В некоторых ситуациях более предпочтительной является схема пассивных измерений.

Пассивные измерения основаны на измерениях характеристик *реального* трафика (рис. 5.5).

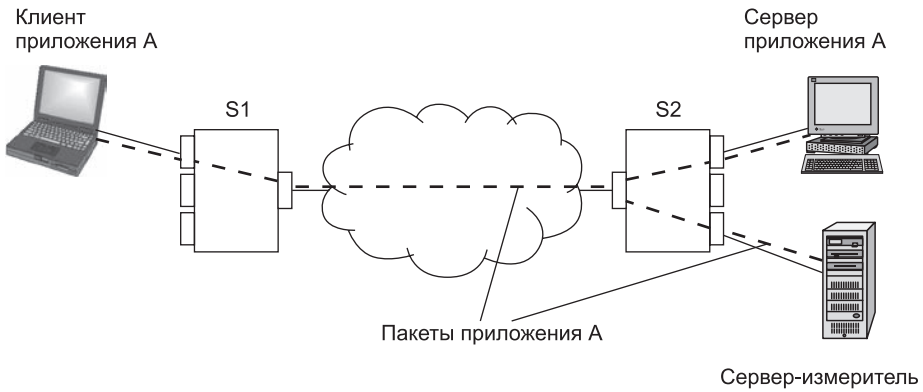


Рис. 5.5. Схема пассивных измерений

Приводя аргументы в пользу схемы активных измерений, мы, в сущности, описали проблемы, которые приходится решать при использовании схемы пассивных измерений: сложности синхронизации клиента и сервера, дополнительные и неопределенные задержки, вносимые универсальными мультипрограммными операционными системами этих компьютеров, отсутствие в заголовке используемых приложением пакетов поля для переноса по сети временно́й отметки.

Частично эти проблемы решаются за счет применения отдельного *сервера-измерителя*. Этот сервер принимает тот же поток пакетов, что и один из узлов, участвующий в обмене пакетами (на рисунке показан случай, когда сервер-измеритель ставится в параллель с сервером приложения *A*). Чтобы сервер-измеритель получал тот же входной поток пакетов, что и оригинальный узел, обычно прибегают к дублированию измеряемого трафика на порт, к которому подключен сервер-измеритель. Такую функцию, называемую **зеркализацией портов**, поддерживают многие коммутаторы локальных сетей. Сервер-измеритель может работать под управлением специализированной операционной системы, оптимизированной для выполнения точных измерений временны́х интервалов.

Сложнее решить проблему синхронизации. Некоторые протоколы переносят временны́е отметки в своих служебных полях, так что если, например, приложение А использует такой протокол, то часть проблемы решается. Однако и в этом случае остается открытым вопрос о точности системного времени в компьютере клиента приложения А; скорее всего, эта точность невысока. Поэтому в пассивном режиме измеряют те характеристики, которые не требуют синхронизации передатчика и приемника, например, оценивают долю потерянных пакетов.

Возможным вариантом пассивной схемы измерений является отсутствие выделенного сервера-измерителя. Некоторые приложения сами выполняют измерения задержек поступающих пакетов, например, такими функциями обладают многие приложения IP-телефонии и видеоконференций, так как информация о задержках пакетов помогает определить возможную причину неудовлетворительного качества работы приложения.

Характеристики задержек пакетов

Для оценки производительности сети используются различные характеристики задержек, в том числе:

- односторонняя задержка пакетов;
- вариация задержки пакета;
- время реакции сети;
- время оборота пакета.

Единичное значение **односторонней задержки пакета** (One-Way Delay Metric, OWD) определенного типа — это интервал времени между моментом помещения в исходящую линию связи *первого* бита пакета узлом-отправителем и моментом приема *последнего* бита пакета с входящей линии связи узла-получателя.

Под определенным типом пакета понимается пакет, который имеет некоторый заранее определенный набор признаков; ими могут быть, например, размер пакета, тип приложения, сгенерировавшего пакет, тип протокола транспортного уровня, который доставил пакет, а также некоторые другие.

Так как в этом определении учитывается время буферизации (сериализации) пакета узлом-получателем, то задержка зависит от размера пакета, и для получения сопоставимых результатов желательно в определении типа пакетов задавать определенный размер пакета. Определение времени задержки с учетом буферизации упрощает измерение времени прихода пакета, так как программно его можно измерить только после завершения записи всего пакета в буфер операционной системы. Кроме того, при получении только одного первого бита пакета невозможно понять, относится ли пакет к интересующему типу.

Для некоторых приложений очень важна *вариация* задержки пакета, которую также называют **джиттером** (jitter). Так, при воспроизведении видеоклипа сама по себе задержка не очень существенна, например, если все пакеты задерживаются ровно на десять секунд, то качество воспроизведения не пострадает, а тот факт, что картинка появляется чуть позже, чем ее отослал сервер, даже не будет замечен пользователем (хотя в интерактивных видеоприложениях, таких как видеоконференции, подобная задержка будет, конечно, уже ощутимо раздражать). А вот если задержки постоянно изменяются в пределах от нуля до 10 секунд, то качество воспроизведения клипа заметно ухудшится, и для компенсации таких переменных задержек нужна предварительная буферизация поступающих пакетов в течение времени, превышающего вариацию задержки.

Единичное значение **вариации задержки** определяется стандартом как разность односторонних задержек для пары пакетов определенного типа, полученных на интервале измерений *T*.

Как и для односторонней задержки, тип пакета может задаваться любыми признаками, при этом размеры обоих пакетов должны быть одинаковыми. Выбор пары пакетов на интервале измерения T должен осуществляться в соответствии с некоторым заранее принятым правилом. Например, пары могут образовываться из всех последовательных пакетов, полученных на интервале; другим примером является выбор пакетов с определенными номерами в последовательности полученных пакетов, например пакетов с номерами 1, 5, 10, 15 и т. д.

Время реакции сети определяется как интервал времени между отправкой запроса пользователя к какой-либо сетевой службе и получением ответа на этот запрос.

Время реакции сети представляет собой интегральную характеристику производительности сети с точки зрения пользователя. Именно эту характеристику имеет в виду пользователь, когда говорит: «Сегодня сеть работает медленно». Время реакции можно представить в виде нескольких слагаемых, например (рис. 5.6):

- времени подготовки запросов на клиентском компьютере ($t_{клиент1}$);
- времени передачи запросов между клиентом и сервером через сеть ($t_{сеть}$);
- времени обработки запросов на сервере ($t_{сервер}$);
- времени передачи ответов от сервера клиенту через сеть (снова $t_{сеть}$);
- времени обработки получаемых от сервера ответов на клиентском компьютере ($t_{клиент2}$).

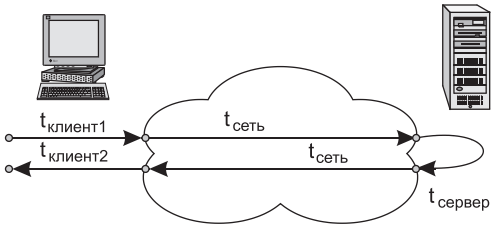


Рис. 5.6. Время реакции и время оборота

Время реакции сети характеризует сеть в целом, в том числе качество работы аппаратного и программного обеспечения серверов. Чтобы отдельно оценить транспортные возможности сети, используется другая характеристика — время оборота данных по сети.

Время оборота пакета (Round Trip Time, **RTT**) определяется как интервал времени между отправкой *первого* бита пакета определенного типа узлом-отправителем узлу-получателю и получением *последнего* бита этого пакета узлом-отправителем после того, как пакет был получен узлом-получателем и отправлен обратно.

Время оборота пакета является составляющей времени реакции сети — это «чистое» время транспортировки данных от узла отправителя до узла назначения и обратно без учета времени, затраченного узлом назначения на подготовку ответа:

$$\mathrm{RTT} = 2 \times t_{сеть}.$$

RTT является удобной для измерений характеристикой, так как для ее получения не требуется синхронизация узла-отправителя и узла-получателя: узел-отправитель ставит

временну́ю отметку на отправляемый пакет, а затем по прибытии его от узла-получателя сравнивает эту отметку со своим текущим системным временем. Однако информативность времени оборота меньше, чем односторонней задержки, так как информация о задержке в каждом направлении теряется, а это может затруднить поиск проблемного пути в сети.

Для каждой из рассмотренных здесь характеристик, как правило, вычисляются их статистические оценки — максимальные, минимальные и средние значения, медианы, квантили и др.

Характеристики скорости передачи

Скорость передачи данных (information rate) измеряется на каком-либо промежутке времени и вычисляется как частное от деления объема переданных за этот период данных на продолжительность периода. Она измеряется в кадрах, пакетах, битах или байтах в секунду. Данная характеристика всегда по определению является *средней* скоростью передачи данных. Однако в зависимости от величины интерва́ла, на котором измеряется скорость, для этой характеристики традиционно используется одно из двух наименований: средняя или мгновенная скорость.

Средняя скорость передачи данных (Sustained Information Rate, SIR) — это среднесрочная характеристика. Она определяется на относительно большом периоде времени, достаточном, чтобы можно было говорить об устойчивом поведении такой случайной величины, которой является скорость. Средняя скорость должна использоваться в паре с параметром, оговаривающим *период контроля* (период измерения) этой величины, например 10 секунд. Это означает, что скорость информационного потока вычисляется каждые 10 секунд. Если бы такие контрольные измерения не проводились, то это лишило бы пользователя возможности предъявлять претензии поставщику в некоторых конфликтных ситуациях. Например, если поставщик в один из дней месяца вообще не будет передавать пользовательский трафик, а в остальные дни разрешит пользователю превышать оговоренный предел, то средняя скорость за месяц окажется в норме. В этой ситуации только регулярный контроль скорости поможет пользователю отстоять свои права.

Мгновенная скорость передачи данных (Instantaneous Information Rate, IIR) — это краткосрочная характеристика, равная средней скорости на очень коротком интервале времени.

Мгновенная скорость, измеренная на *интервале передачи пакета*, равна битовой скорости протокола физического уровня (например, для пакета стандарта 1Gigabit Ethernet она равна 1 Гбит/с) и равна нулю в интервалах между передачами пакетов. Все пакеты, передаваемые по каналам связи одного и того же стандарта, имеют *одну и ту же мгновенную скорость*. Эта скорость — постоянная величина, она не зависит ни от объема пакета, ни от характеристик потока, к которому он относится; не может быть ни уменьшена, ни увеличена.

ПРИМЕЧАНИЕ

Заметим, что выше мы говорили о скорости передачи **отдельного** пакета. Другое дело — скорость **потока** пакетов. В этом случае измерение скорости выполняется на относительно большом интервале времени (большем, чем время передачи одного пакета), в течение этого времени может передаваться разное, но, как правило, большое количество пакетов. Как следует из определения скорости, чем больше объем передаваемых данных (количество переданных пакетов), тем выше скорость. И чем больше интервалы между пакетами, тем ниже скорость. Очевидно, что в отличие от скоростей отдельных пакетов, потоки данных могут передаваться по одному и тому же каналу с разными скоростями, но при этом скорости потоков не превосходят пропускную способность канала.

Вариация мгновенной скорости, то есть степень отклонения мгновенных значений от средней скорости, характеризует неравномерность трафика.

На рис. 5.7 показан реальный график изменения скорости выходного трафика, передаваемого от университетского кампуса к серверам провайдера в течение рабочего дня (нижний график описывает входной трафик). На графике имеются участки, имеющие форму *пиков*, на которых мгновенная скорость значительно отличается от средней. Такие участки называют **пульсациями**. На рисунке отмечены *пульсация 1* и *пульсация 2*. Каждая из них характеризуется **периодом пульсации** и пиковой скоростью.

Пиковая скорость передачи данных (Peak Information Rate, **PIR**) — это средняя скорость на периоде пульсации. Пиковая скорость является краткосрочной характеристикой, она отражает способность сети справляться с пиковыми нагрузками, характерными для пульсирующего трафика и приводящими к перегрузке. Если период усреднения в определении пиковой скорости выбран достаточно небольшим, то можно говорить, что пиковая скорость является заданным верхним пределом для мгновенной скорости пользовательского потока.



Рис. 5.7. Скорость передачи трафика. Пульсации

Величина пульсации (обычно обозначаемая *B* от англ. *B*urst) служит для оценки емкости буфера коммутатора, необходимого для хранения данных во время перегрузки. Величина пульсации равна общему объему данных, поступающих на коммутатор в течение периода пульсации *T* передачи данных с пиковой скоростью (PIR): $B = \text{PIR} \times T$.

Еще одной характеристикой неравномерности передачи данных является **коэффициент пульсации трафика** — это отношение максимальной пиковой скорости к средней скорости трафика, измеренной за длительный период времени. Неопределенность временны́х периодов делает коэффициент пульсации *качественной* характеристикой трафика.

Скоростные характеристики процесса передачи данных играют важную роль в отношениях поставщика услуг с клиентами. В *соглашении SLA об уровне услуг* провайдер указывает ограничения на трафик, передаваемый клиентом, в виде максимально допустимой скорости на заданных интервалах усреднения. Иногда в соглашении SLA фигурирует один такой период, иногда больше — тогда речь идет об ограничениях на скорости, вычисленных по-разному, например, на продолжительном и на коротком периодах усреднения, что позволяет более точно описать процесс передачи пульсирующего компьютерного

трафика через сеть. На рис. 5.8 показаны ограничения провайдера на **максимально допустимую среднюю скорость** и **максимально допустимую пиковую скорость** пользовательского трафика.

Скорость передачи данных можно измерять между любыми двумя узлами, или точками, сети, например, между клиентским компьютером и сервером, между входным и выходным портами маршрутизатора.

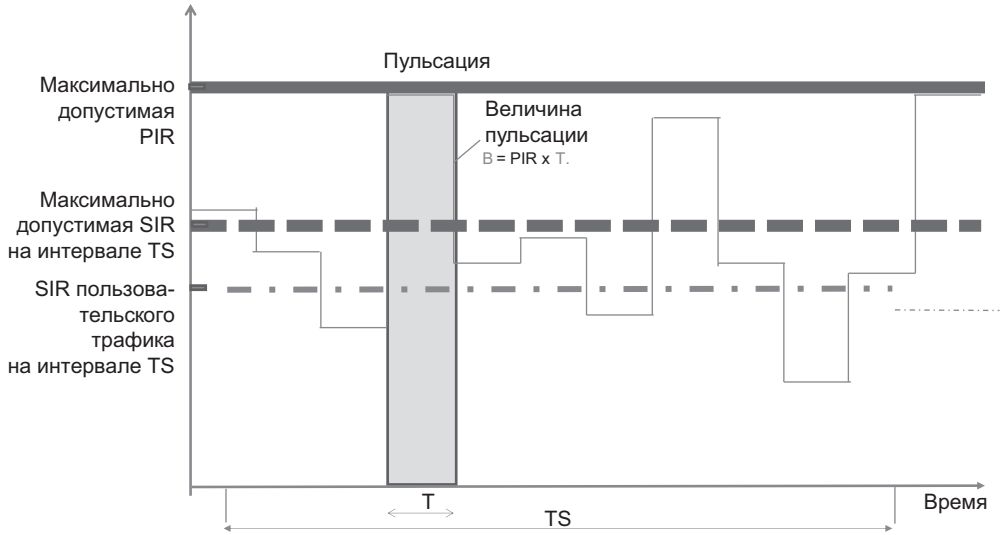


Рис. 5.8. Ограничения провайдера на скорость пользовательского трафика

Характеристики надежности сети

В том случае, если пакет не прибыл в узел назначения за некоторое достаточно большое время (точное значение определяет разработчик системы измерений), пакет считается утерянным.

В качестве характеристики потерь пакетов используется **доля потерянных пакетов** (обозначим ее L), равная отношению количества потерянных пакетов (N_L) к общему количеству переданных пакетов (N):

$$L = N_L/N.$$

Может использоваться и аналогичная характеристика, оперирующая не количествами потерянных и переданных пакетов, а объемами данных, содержащихся в этих пакетах.

Для описания надежности отдельных устройств служат такие показатели надежности, как **среднее время наработки на отказ**, **вероятность отказа**, **интенсивность отказов**. Однако эти показатели пригодны только для оценки надежности простых элементов и устройств, которые при отказе любого своего компонента переходят в неработоспособное состояние. Сложные системы, состоящие из многих компонентов, могут при отказе одного из компонентов сохранять свою работоспособность. В связи с этим для оценки надежности сложных систем применяется другой набор характеристик.

Доступность (availability) означает долю времени, в течение которого система или служба находится в работоспособном состоянии.

Доступность является долговременной статистической характеристикой, поэтому измеряется за большой промежуток времени, которым может быть день, месяц или год. Примером высокого уровня доступности является коммуникационное оборудование телефонных сетей, лучшие представители которого обладают так называемой доступностью «пять девяток». Это означает, что доступность равна 0,99999, что соответствует чуть более 5 минутам простоя в год. Заметим, что существующее оборудование и услуги передачи данных только стремятся к такому рубежу, но рубеж трех девяток уже достигнут. Доступность услуги является универсальной характеристикой, которая важна как пользователям, так и поставщикам услуг.

Еще одной характеристикой надежности сложных систем является **отказоустойчивость** (fault tolerance). Под отказоустойчивостью понимается способность системы скрывать от пользователя отказ отдельных ее элементов.

Например, если коммутатор оснащен двумя коммутационными центрами, работающими параллельно, то отказ одного их них не приведет к полной остановке коммутатора. Однако производительность коммутатора снизится, он будет обрабатывать пакеты вдвое медленнее. В отказоустойчивой системе отказ одного из ее элементов приводит к некоторому снижению качества ее работы — **деградации**, а не к полной ее остановке. В качестве еще одного примера можно назвать использование двух физических каналов для соединения коммутаторов. В нормальном режиме работы трафик передается по двум каналам со скоростью C Мбит/с. При отказе одного из них трафик будет продолжать передаваться, но уже со скоростью $C/2$ Мбит/с. Однако из-за того, что во многих случаях количественно определить степени деградации системы или услуги достаточно сложно, отказоустойчивость чаще всего применяется как качественная характеристика.

Характеристики сети поставщика услуг

Рассмотрим основные характеристики, которыми оперирует поставщик услуг, оценивая эффективность своей сети. Эти характеристики — расширяемость, масштабируемость, управляемость и совместимость — являются качественными, то есть не могут быть выражены числами и соотношениями.

Термины «расширяемость» и «масштабируемость» иногда используют как синонимы, что неверно. Принципиальные смысловые различия заключаются в следующем.

Расширяемость означает возможность сравнительно простого добавления отдельных компонентов сети (пользователей, компьютеров, приложений, служб), наращивания длины сегментов кабелей и замены существующей аппаратуры более мощной.

При этом принципиально важно, что простота расширения системы иногда может обеспечиваться в *определенных пределах*. Например, локальная сеть Ethernet, построенная на основе одного разделяемого сегмента коаксиального кабеля, обладает хорошей расширяемостью в том смысле, что позволяет легко подключать новые станции. Однако такая сеть имеет ограничение на число станций, которое не должно превышать 30–40. Хотя сеть допускает физическое подключение к сегменту и большего числа станций (до 100), при

этом резко снижается производительность сети. Наличие такого ограничения и является признаком плохой масштабируемости системы при ее хорошей расширяемости.

Масштабируемость означает, что сеть позволяет наращивать количество узлов и протяженность связей в очень широких пределах, при этом производительность сети не снижается.

Для обеспечения масштабируемости сети приходится применять дополнительное коммуникационное оборудование и специальным образом структурировать сеть. Обычно масштабируемое решение обладает многоуровневой иерархической структурой, которая позволяет добавлять элементы на каждом уровне иерархии без изменения главной идеи проекта. Примером хорошо масштабируемой сети является интернет, технология которого (TCP/IP) оказалась способной поддерживать сеть глобально, то есть в масштабах земного шара.

Не только сама сеть должна быть масштабируемой, но и устройства, работающие на магистрали сети, также должны обладать этим свойством, так как рост сети не должен приводить к необходимости постоянной смены оборудования. Поэтому магистральные коммутаторы и маршрутизаторы строятся обычно по модульному принципу, позволяя наращивать количество интерфейсов и производительность обработки пакетов в широких пределах.

Управляемость сети подразумевает возможность централизованно контролировать состояние основных элементов сети, выявлять и разрешать проблемы, возникающие при работе сети, анализировать производительность и планировать развитие сети.

Управляемость предполагает наличие в сети некоторых автоматизированных *средств администрирования*, которые взаимодействуют с программным и аппаратным обеспечением сети с помощью коммуникационных протоколов. В идеале средства администрирования сети обеспечивают *наблюдение и контроль* за каждым элементом сети и, обнаружив проблему, активизируют определенное действие, например, исправляют ситуацию и уведомляют администратора о том, что произошло и какие шаги предприняты. Одновременно с этим система администрирования должна *накапливать данные*, на основании которых можно планировать развитие сети. Наконец, система администрирования должна быть независима от производителя и обладать удобным интерфейсом, позволяющим выполнять все действия с одной консоли.

Решая тактические задачи, администраторы и технический персонал сталкиваются с ежедневными проблемами поддержания работоспособности сети. Эти задачи требуют быстрого решения, обслуживающий сеть персонал должен оперативно реагировать на сообщения о неисправностях, поступающие от пользователей или автоматических средств администрирования сети. Постепенно становятся заметными более общие проблемы производительности, конфигурирования сети, обработки сбоев и безопасности данных, требующие стратегического подхода, то есть *планирования* сети. Планирование, кроме того, подразумевает умение прогнозировать изменения в требованиях пользователей к сети, решение вопросов о применении новых приложений, новых сетевых технологий и т. п.

Полезность систем администрирования особенно ярко проявляется в больших сетях: корпоративных или публичных глобальных. Без систем администрирования в таких сетях потребовалось бы содержание огромного штата обслуживающего персонала.

Однако в настоящее время большинство существующих средств вовсе не управляет сетью, а всего лишь обеспечивает *наблюдение* за ее работой и *фиксацию* важных событий, например отказов устройств. Реже системы администрирования выполняют активные

действия, автоматически ликвидируя последствия нежелательного события или предотвращая его.

Совместимость, или **интегрируемость**, сети означает, что сеть способна включать в себя самое разнообразное программное и аппаратное обеспечение, то есть в ней могут сосуществовать различные операционные системы, поддерживающие разные стеки коммуникационных протоколов, а также аппаратные средства и приложения от разных производителей. Сеть, состоящая из разнотипных элементов, называется неоднородной или *гетерогенной*, а если гетерогенная сеть работает без проблем, то она является *интегрированной*. Основной путь построения интегрированных сетей — использование модулей, выполненных в соответствии с открытыми стандартами и спецификациями.

Приложения и качество обслуживания

Поскольку существующие приложения в общем случае предъявляют разные требования к качеству обслуживания, важной задачей является их классификация в этом отношении. В качестве критериев классификации будем использовать следующие свойства приложений:

- степень равномерности порождаемого ими трафика;
- чувствительность приложений к задержкам пакетов;
- чувствительность приложений к потерям и искажениям пакетов.

Степень равномерности порождаемого трафика

В отношении равномерности порождаемого трафика приложения делятся на два класса: приложения с потоковым трафиком и приложения с пульсирующим трафиком.

Приложения с потоковым трафиком (stream) порождают равномерный поток данных, который поступает в сеть с **постоянной битовой скоростью** (Constant Bit Rate, CBR). В случае коммутации пакетов трафик таких приложений представляет собой последовательность пакетов одинакового размера (равного B бит), следующих друг за другом через один и тот же интервал времени T (рис. 5.9, *а*).

CBR потокового трафика может быть вычислена путем усреднения на интервале:

$$\text{CBR} = B/T \text{ бит/с}.$$

Приложения с пульсирующим трафиком (burst) отличаются высокой степенью непредсказуемости, в этих приложениях периоды молчания сменяются пульсациями, в течение которых пакеты «плотно» следуют друг за другом. В результате трафик характеризуется **переменной битовой скоростью** (Variable Bit Rate, VBR), что иллюстрирует рис. 5.9, *б*. Так, при работе приложений файлового сервиса интенсивность трафика, генерируемого приложением, может падать до нуля, когда файлы не передаются, и повышаться до максимально доступной интенсивности, ограниченной только возможностями сети, когда файловый сервер передает файл.

На рисунке показана ситуация, когда за период длительностью $5T$ передано три пакета (все пакеты имеют одинаковый размер B), затем (за период длительностью T) передано 5 пакетов, а за период длительностью $6T$ — 2 пакета. *Пиковой скоростью* трафика является скорость за второй период, когда за время T было передано 5 пакетов, поэтому PIR = $5B/T$.

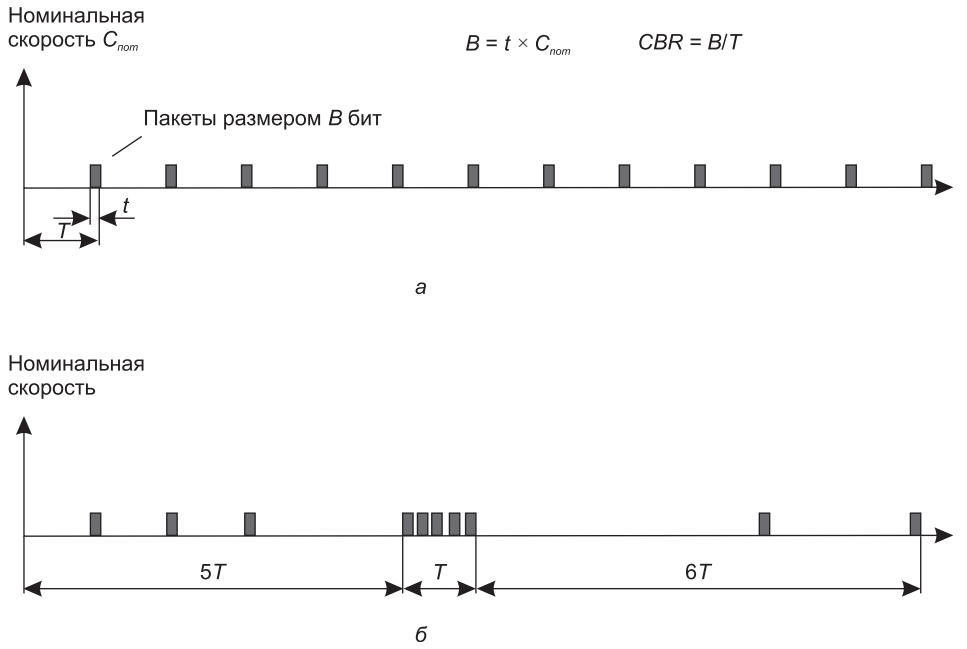


Рис. 5.9. Потоковый (*а*) и пульсирующий (*б*) трафики

В то же время *средняя скорость* передачи данных (Sustained Information Rate, SIR) за все периоды наблюдений составила $10B/12T = 5B/6T$.

Для приведенного примера можно подсчитать *коэффициент пульсации*. Он равен отношению пиковой скорости к средней скорости трафика, измеренной за длительный период времени. Таким образом, коэффициент пульсации равен PIR/SIR = $(5B/T)/(5B/6T) = 6$.

Практически любой трафик, даже трафик потоковых приложений, имеет ненулевой коэффициент пульсации. Просто значения коэффициентов пульсации у потокового и пульсирующего трафиков существенно различаются. У приложений с пульсирующим трафиком он обычно находится в пределах от 2 до 100, а у потоковых приложений он близок к 1. В локальных сетях коэффициент пульсации обычно выше, чем в глобальных, поскольку на магистралях глобальных сетей трафик представляет собой сумму трафиков многих источников, что по закону больших чисел приводит к сглаживанию результирующего трафика.

Чувствительность приложений к задержкам пакетов

Еще один критерий классификации приложений — их чувствительность к задержкам пакетов и их вариациям. Далее перечислены основные типы приложений в порядке повышения чувствительности к задержкам пакетов:

- **асинхронные приложения** практически не имеют ограничений на время задержки (*эластичный трафик*). Пример такого приложения — электронная почта;

- **интерактивные приложения**. Задержки могут быть замечены пользователями, но они не сказываются негативно на функциональности приложений. Пример — текстовый редактор, работающий с удаленным файлом;
- **изохронные приложения** имеют порог чувствительности к вариациям задержек, при превышении которого резко снижается функциональность приложений. Пример — передача голоса, когда при превышении порога вариации задержек в 100–150 мс искажение голоса становится неприемлемым;
- **сверхчувствительные к задержкам приложения**, для которых задержка доставки данных сводит их функциональность к нулю. Пример — приложения, управляющие техническим объектом в *реальном времени*. При запаздывании управляющего сигнала на объекте может произойти авария.

Вообще говоря, *интерактивность* приложения всегда повышает его чувствительность к задержкам. Например, широковещательная рассылка аудиоинформации может выдерживать значительные задержки в передаче пакетов (оставаясь чувствительной к вариациям задержек), а интерактивный телефонный или телевизионный разговор их не терпит, что хорошо заметно при трансляции разговора через спутник. Длительные паузы в разговоре вводят собеседников в заблуждение, часто они теряют терпение и начинают очередную фразу одновременно.

Наряду с приведенной классификацией, тонко дифференцирующей чувствительность приложений к задержкам и их вариациям, существует и более грубое деление приложений по этому признаку на два класса: асинхронные и синхронные. К **асинхронным** относят те приложения, которые нечувствительны к задержкам передачи данных в очень широком диапазоне, вплоть до нескольких секунд, а все остальные приложения, на функциональность которых задержки влияют существенно, относят к **синхронным** приложениям. Интерактивные приложения могут относиться как к асинхронным (например, текстовый редактор), так и к синхронным (например, видеоконференция).

Чувствительность приложений к потерям и искажениям пакетов

И наконец, последним критерием классификации приложений является их чувствительность к потерям пакетов. В этой связи приложения обычно делят на две группы.

- **Приложения, чувствительные к потере данных.** Практически все приложения, передающие алфавитно-цифровые данные (к которым относятся текстовые документы, коды программ, числовые массивы и т. п.), обладают высокой чувствительностью к потере отдельных, даже небольших фрагментов данных. Такие потери часто ведут к полному обесцениванию остальной успешно принятой информации. Например, отсутствие хотя бы одного байта в коде программы делает ее совершенно неработоспособной. Все традиционные сетевые приложения (файловый сервис, сервис баз данных, электронная почта и т. д.) относятся к этому типу приложений.
- **Приложения, устойчивые к потере данных.** К этому типу относятся многие приложения, передающие трафик с информацией об инерционных физических процессах. Устойчивость к потерям объясняется тем, что небольшое количество отсутствующих данных можно определить на основе принятых. Так, при потере одного пакета, несущего

несколько последовательных замеров голоса, отсутствующие замеры при воспроизведении голоса могут быть заменены аппроксимацией на основе соседних значений. К такому типу относится большая часть приложений, работающих с мультимедийным трафиком (аудио- и видеоприложения). Однако устойчивость к потерям имеет свои пределы, поэтому процент потерянных пакетов не может быть большим (например, не более 1 %). Можно отметить также, что не любой мультимедийный трафик устойчив к потерям данных. Так, компрессированный голос и видеоизображение очень чувствительны к потерям, поэтому относятся к первому типу приложений.

Методы обеспечения качества обслуживания

Методы обеспечения **качества обслуживания** (Quality of Service, **QoS**) занимают сегодня важное место в арсенале технологий сетей с коммутацией пакетов, так как они обеспечивают устойчивую работу современных мультимедийных приложений — таких, как IP-телефония, видео- и радиовещание, интерактивное дистанционное обучение и т. п.

В методах обеспечения качества обслуживания используются различные механизмы, позволяющие снизить негативные последствия временных перегрузок, возникающих в сетях с коммутацией пакетов. Эти механизмы делятся на:

- **средства управления перегрузкой** (congestion management), которые начинают работать, когда сеть уже перегружена; к ним относится управление очередями;
- **средства предотвращения перегрузок** (congestion avoidance), которые, как следует из названия, предпринимают превентивные меры для предотвращения перегрузок; к такого рода средствам относится кондиционирование трафика, обратная связь, резервирование ресурсов и трафик-инжиниринг.

Очереди являются неотъемлемым атрибутом сетей с коммутацией пакетов. Сам принцип работы таких сетей подразумевает наличие буфера во входных и выходных интерфейсах коммутирующих устройств. Именно буферизация пакетов во время перегрузок является основным механизмом, обеспечивающим *высокую производительность* сетей этого типа.

В то же время очереди означают *снижение качества обслуживания* из-за неопределенности задержек и потерь пакетов при передаче данных через сеть из-за переполнения отведенного под очередь буфера коммутатора или маршрутизатора.

Таким образом, операторам пакетных сетей, весьма заинтересованным в передаче пульсирующего трафика, необходимы средства достижения *компромисса* между как можно более высокой загрузкой своей сети и требуемым клиентами качеством обслуживания.

В методах QoS используется широкий набор механизмов, направленных на снижение негативных последствий пребывания пакетов в очередях с сохранением в то же время положительной роли очередей. Большинство из них учитывает факт существования в сети трафика различного типа, а именно то, что каждый тип трафика предъявляет разные требования к характеристикам производительности и надежности сети. Однако добиться одновременного соблюдения *всех* характеристик QoS для *всех* видов трафика весьма сложно.

Одним из наиболее значимых факторов, влияющих на характеристики качества обслуживания, является уровень загрузки сети трафиком, то есть *уровень использования пропускной*

способности линий связи сети. Напомним, что пропускная способность — это характеристика физического канала, которая равна максимально возможной скорости передачи информации по этому каналу.

Пропускную способность сети изменить непросто, поскольку она определяется быстродействием интерфейсов коммуникационного оборудования и качеством линий связи, их соединяющих. Повышение пропускной способности сети — это дорогостоящая операция, связанная с заменой оборудования, которую операторы сетей проводят не очень часто (как правило, раз в несколько лет).

Если уровень использования пропускной способности постоянно является достаточно низким, то трафик всех приложений обслуживается с высоким качеством большую часть времени (хотя кратковременные перегрузки сети, приводящие к задержкам и потерям пакетов, все равно возможны, но они случаются редко). Такое состояние сети называется «недогруженным», или же используется термин **«сеть с избыточной пропускной способностью»** (англоязычный термин overprovisioning). Постоянно поддерживать все части сети в недогруженном состоянии весьма дорого, но для наиболее ответственной части сети, такой как магистраль, подобный подход применяется и связан с постоянным слежением за уровнем загрузки каналов магистрали и обновлением оборудования с более высокой пропускной способностью по мере приближения загрузки к критическому уровню. Методы QoS основаны на другом подходе.

> Идея, лежащая в основе всех методов поддержания характеристик QoS, заключается в следующем: общая производительность каждого ресурса должна делиться между разными классами трафика дифференцированно, с учетом специфических требований приложений разного класса.

Очевидно, что эти методы усложняют сетевые устройства, которые теперь должны «знать» требования всех классов трафика, уметь их распознавать и распределять пропускную способность сети между ними. Последнее свойство обычно достигается за счет использования для каждого выходного интерфейса коммуникационного устройства нескольких очередей пакетов вместо одной очереди; при этом в очередях применяют различные алгоритмы обслуживания пакетов, чем и достигается дифференцированное обслуживание трафика различных классов. Поэтому методы QoS часто ассоциируются с *техникой управления очередями.*

Помимо собственно техники организации очередей, к методам QoS относят методы контроля параметров потока трафика, так как для гарантированно качественного обслуживания нужно быть уверенными, что обслуживаемые потоки соответствуют набору ограничений, определенному для них. Эта группа методов QoS получила название *методов кондиционирования трафика.*

Особое место занимают *методы обратной связи,* которые предназначены для уведомления источника трафика о перегрузке сети. Эти методы рассчитаны на то, что при получении уведомления источник снизит скорость выдачи пакетов в сеть и тем самым ликвидирует причину перегрузки.

К методам QoS тесно примыкают *методы инжиниринга трафика.* Согласно методам инжиниринга трафика маршруты передачи данных управляются таким образом, чтобы обеспечить сбалансированную загрузку всех ресурсов сети и исключить за счет этого перегрузку коммуникационных устройств и образование длинных очередей.

Управление очередями

Пользователь формулирует свои требования к качеству обслуживания в виде некоторых предельных значений характеристик QoS, которые не должны быть превышены — например, он может указать, что предельное значение вариации задержки пакетов не должно превышать 50 мс с вероятностью 0,99. Но как заставить сеть справиться с поставленной задачей? Какие меры нужно предпринять, чтобы вариации задержек действительно не превысили эту величину? Для понимания механизмов поддержки QoS полезно исследовать процесс образования очередей на сетевых устройствах. В главе 3 (раздел «Буферизация пакетов») мы уже обсудили некоторые причины возникновения очередей в коммутирующих устройствах.

Давайте еще раз обратимся к рис. 3.8, на котором показана простейшая схема коммутатора (маршрутизатора). Центральным элементом устройства является *коммутирующий блок*, который передает данные между тремя своими интерфейсами. Если интенсивности входных потоков и производительность коммутирующего блока таковы, что последний не всегда успевает обработать поступающие пакеты, то избыточные пакеты помещают для ожидания обслуживания во *входные очереди* (буферы). Когда коммутирующий блок освобождается, специальный блок коммутатора — *арбитр входных очередей* (не показанный на рисунке), следуя определенному для него правилу, выбирает очередной пакет из очереди и направляет его на обработку в коммутирующий блок. Таким правилом может быть, например, FIFO — «первый пришел — первый ушел» — или более сложный алгоритм с учетом приоритетов.

Пакеты, прошедшие обработку в коммутирующем блоке, направляются на один из назначенных для них *выходных интерфейсов*. Из-за случайного характера интервалов между пакетами потока и ограниченного значения пропускной способности выходного интерфейса может возникнуть ситуация перегрузки, когда в момент пульсации трафика очередной пакет не будет принят выходным интерфейсом на передачу. Следовательно, необходимо предусмотреть перед каждым выходным интерфейсом буфер — *выходную очередь* — для сохранения пакетов на время перегрузки. Пакеты выбираются из выходной очереди и направляются на связанный с ней выходной интерфейс коммутатора в соответствии с правилом, которым руководствуется *арбитр выходных очередей*.

Размер буфера сетевого устройства определяет максимальную длину очереди ожидающих обслуживания пакетов, а если пакет поступает при заполненном буфере, то он просто отбрасывается.

Анализ очередей

Существует ветвь прикладной математики, предметом которой являются процессы образования очередей. Эта дисциплина так и называется — **теория очередей**. Мы не будем углубляться в математические основы этой теории, обратив внимание читателей только на некоторые ее выводы, существенные для рассматриваемой нами проблемы QoS.

Теория очередей рассматривает временны́е процессы образования очередей в буфере абстрактного устройства, в который поступает случайный поток абстрактных **заявок** на обслуживание. Модели теории очередей позволяют оценить среднюю длину очереди в буфере и среднее время ожидания заявки в очереди в зависимости от характеристик входного потока и времени обслуживания.

При применении теории очередей к анализу процессов, происходящих в компьютерных сетях, заявками на обслуживание являются пакеты данных, а разделяемыми **обслуживающими устройствами** — различные программы, сетевые устройства, их выходные интерфейсы. На рис. 5.10 показана модель одного из выходных интерфейсов коммутатора.

Показанный на рис. 5.10 входной поток заявок соответствует потоку пакетов между коммутирующим блоком и одним из выходных интерфейсов коммутатора, как это было показано ранее на рис. 3.8. Этот поток заявок характеризуется *средней интенсивностью* λ, то есть средний интервал между заявками равен $T=1/\lambda$. Средняя интенсивность потока заявок является аналогом средней скорости трафика, поступающего на выходной интерфейс. Заявки направляются к обслуживающему устройству, но если оно занято обслуживанием другой заявки, то их помещают в **очередь**, в которой они ждут, пока не освободится устройство. Обслуживающее устройство обрабатывает каждую заявку в среднем в течение времени b, при этом *средняя интенсивность обработки* потока заявок составляет $\mu=1/b$. Очевидно, что параметр модели μ является аналогом пропускной способности интерфейса.

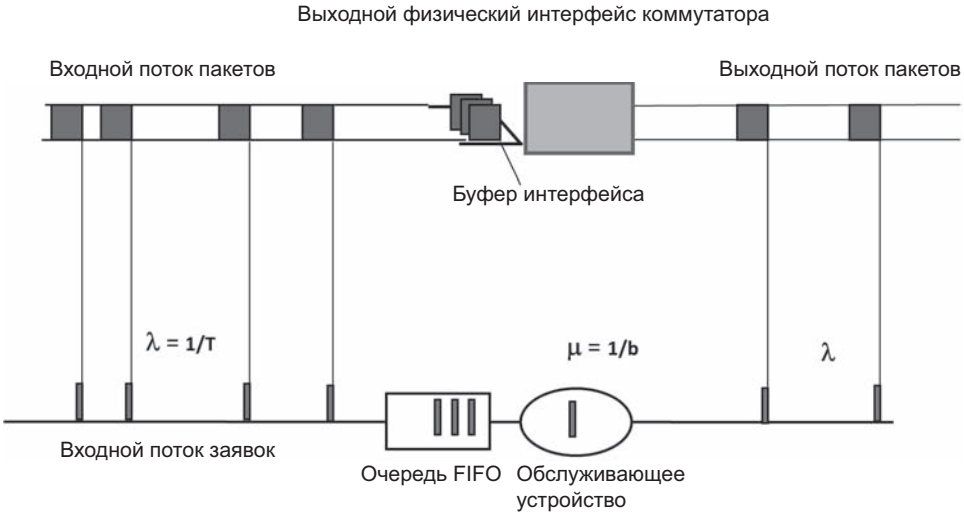


Рис. 5.10. Модель выходного интерфейса коммутатора как разделяемого ресурса

Модели теории очередей весьма упрощенно описывают процессы, происходящие в реальных устройствах. Тем не менее они полезны для понимания основных факторов, влияющих на величину очереди в буфере сетевого устройства или на время пребывания пакета в буфере.

Одним из таких факторов является **коэффициент загрузки** (использования) обслуживающего устройства. Он равен отношению средней интенсивности λ поступления заявок в устройство к средней интенсивности μ обработки заявок обслуживающим устройством: $\rho = \lambda/\mu$.

В теории массового обслуживания для некоторых типов моделей найдены аналитические зависимости, описывающие среднее время нахождения заявки в очереди W в зависимости от коэффициента загрузки ρ. Одна из таких зависимостей, описывающих систему с одной очередью, обслуживаемой по принципу FIFO, показана на рис. 5.11.

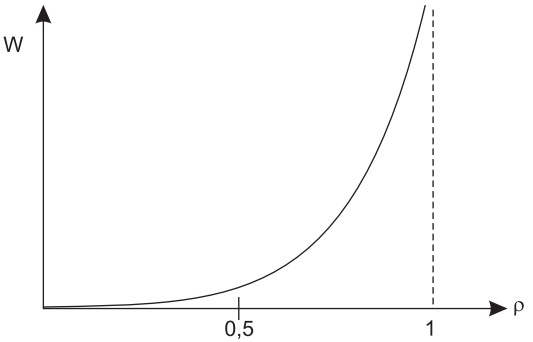


Рис. 5.11. Зависимость среднего времени ожидания заявки от коэффициента загрузки ресурса

Как видно из поведения кривой, коэффициент ρ играет ключевую роль в образовании очереди. Если значение ρ близко к нулю, то среднее время ожидания тоже очень близко к нулю. Это означает, что пакеты почти никогда не ожидают обслуживания в буфере (в момент их прихода он оказывается пустым), а сразу передаются на выход. И наоборот, если ρ приближается к 1, то время ожидания растет очень быстро и нелинейно (и в пределе равно бесконечности). Такое поведение очереди интуитивно понятно, ведь чем ближе средние значения интервалов между пакетами к среднему времени их обслуживания, тем сложнее обслуживающему устройству (интерфейсу) справляться с нагрузкой.

Сетевые инженеры хорошо знакомы с видом графика, представленного на рис. 5.12, так как он соответствует поведению тех кривых, которые инженеры видят при анализе результатов мониторинга задержек и потерь пакетов в реальных сетях, а именно резкому ухудшению качества обслуживания при достижении коэффициента использования пропускной способности интерфейсов сети некоторого порогового значения.

В приведенном графике есть и нечто неожиданное. Трудно представить, что обслуживающее устройство (сетевой ресурс) практически перестает справляться со своими обязанностями, когда его коэффициент использования приближается к 1. Ведь в этом случае нагрузка не превышает его возможностей, а только приближается к этому пределу. Интуитивно не очень понятна также причина существования очередей при значениях ρ в окрестностях 0,5. Интенсивность обработки трафика вдвое превышает интенсивность нагрузки, а очереди существуют! Такие парадоксальные на первый взгляд результаты характерны для систем, в которых протекают случайные процессы. Так как во внимание принимаются *средние* значения интенсивностей потоков на больших промежутках времени, то на небольших промежутках времени они могут *существенно отклоняться* от этих значений.

Существует еще один важный параметр, оказывающий непосредственное влияние на образование очередей в сетях с коммутацией пакетов. Этим параметром является вариация интервалов входного потока пакетов, то есть пульсация входного трафика. На рис. 5.12 показано семейство зависимостей W от ρ, полученных для разных значений *коэффициента вариации CV* интервалов входного потока пакетов. Из рисунка видно, что чем меньше пульсирует входной поток (CV приближается к нулю), тем меньше проявляется эффект лавинообразного образования очереди при приближении коэффициента загрузки ресурса к 1. И наоборот, чем больше CV, тем раньше (при меньших значениях ρ) начинает проявляться этот эффект.

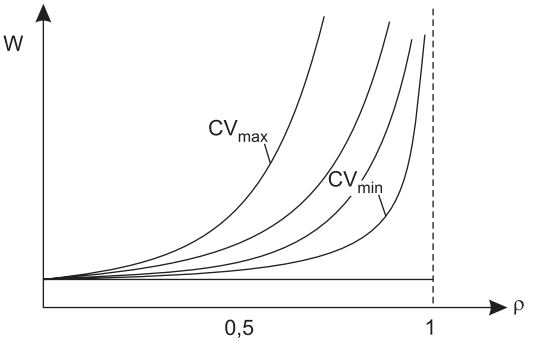


Рис. 5.12. Влияние степени пульсации потока на задержки

Из поведения графиков на рисунке можно сделать следующий вывод:

> Для уменьшения задержек нужно снижать как коэффициент загрузки ресурса, так и пульсацию потока.

Следует подчеркнуть, что модели теории очередей из-за упрощенного представления процессов, протекающих в коммуникационных устройствах сети, дают только *качественную* картину зависимостей задержек и потерь пакетов от параметров трафика и производительности устройств. Более адекватные результаты могут быть получены путем имитационного моделирования, но надежнее всего полагаться на измерение задержек и потерь пакетов в реальной сети с помощью соответствующих тестеров и систем.

Очереди и различные классы трафика

Посмотрим, как можно применить знания о зависимости поведения очередей от коэффициента загрузки для реализации основной идеи методов QoS, а именно дифференцированного обслуживания классов трафика с различными требованиями к характеристикам производительности и надежности сети. Для простоты будем пока делить все потоки на два класса — *чувствительный к задержкам* (трафик реального времени, например голосовой) и *эластичный*, допускающий большие задержки, но чувствительный к потерям данных.

Если обеспечить для чувствительного к задержкам трафика коэффициент загрузки каждого ресурса не более 0,2, то, очевидно, задержки в каждой очереди будут небольшими и, скорее всего, приемлемыми для многих типов приложений этого класса. Для эластичного трафика, слабо чувствительного к задержкам, можно допустить более высокий коэффициент загрузки, но не более 0,9. Чтобы пакеты этого класса не терялись, нужно предусмотреть для них буферную память, достаточную для хранения всех пакетов периода пульсации. Эффект от такого распределения загрузки ресурса иллюстрирует рис. 5.13.

Задержки чувствительного к задержкам трафика равны w_s, а задержки эластичного трафика — w_e.

Чтобы добиться различных коэффициентов использования ресурсов для разных классов трафика, нужно в каждом коммутаторе для каждого ресурса поддерживать две разные очереди. Алгоритм выборки пакетов из очередей должен отдавать предпочтение очереди

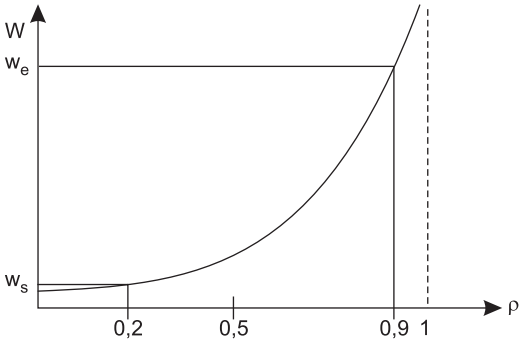


Рис. 5.13. Обслуживание эластичного и чувствительного к задержкам трафика

чувствительных к задержкам пакетов. Если бы все пакеты первой очереди обслуживались приоритетно, а пакеты второй очереди — только тогда, когда первая очередь пуста, то для трафика первой очереди трафик второй очереди фактически перестал бы существовать. Поэтому если отношение средней интенсивности приоритетного трафика λ_1 к производительности ресурса μ равно 0,2, то и коэффициент загрузки для него равен 0,2. Для эластичного трафика, пакеты которого всегда ждут обслуживания приоритетных пакетов, коэффициент загрузки подсчитывается по-другому. Если средняя интенсивность эластичного трафика равна λ_2, то для него ресурс будет загружен на $(\lambda_1 + \lambda_2)/\mu$. Следовательно, если мы хотим, чтобы для эластичного трафика коэффициент загрузки составлял 0,9, то его интенсивность должна вычисляться из соотношения $\lambda_2/\mu = 0{,}7$.

Можно ввести более чем два класса обслуживания и стараться, чтобы каждый класс работал на своей части кривой задержек. Если такая задача решена, то можно обеспечить улучшение характеристик QoS за счет других методов, например, снижая пульсацию трафика. Осталось выяснить, каким образом обеспечить такие условия для разных классов трафика в каждом узле сети.

Техника управления очередями

Управление очередями используется для оптимизации работы устройства в периоды временных перегрузок, когда оно не справляется с передачей пакетов на выходной интерфейс в том темпе, в котором они поступают. Алгоритмы управления очередями не предотвращают перегрузок, а лишь некоторым «справедливым» образом в условиях дефицита перераспределяют ресурсы между различными потоками или классами трафика.

Очередь FIFO

В **очереди FIFO** (First In — First Out) в случае перегрузки все пакеты помещаются в *одну* общую очередь и выбираются из нее в том порядке, в котором поступили, то есть в соответствии с принципом «первый пришел — первый ушел». Во всех устройствах с коммутацией пакетов алгоритм FIFO используется по умолчанию. Достоинствами этого подхода являются простота реализации и отсутствие потребности в конфигурировании. Однако ему присущ и коренной недостаток — *невозможность дифференцированной обработки пакетов различных потоков.* Все пакеты стоят в общей очереди на равных основаниях. Одинаково

обслуживаются как пакеты чувствительного к задержкам голосового трафика, так и нечувствительного к задержкам, но очень интенсивного трафика резервного копирования, длительные пульсации которого могут надолго задержать голосовой пакет.

Приоритетное обслуживание

Очереди с приоритетным обслуживанием очень популярны во многих областях вычислительной техники, в частности, в операционных системах, когда одним приложениям нужно отдать предпочтение перед другими при обработке их в мультипрограммной смеси. Применяются эти очереди и для преимущественной обработки некоторого класса трафика. Механизм приоритетного обслуживания основан на разделении всего сетевого трафика на небольшое количество классов и последующего назначения каждому классу некоторого числового признака — приоритета.

Приоритет — это число, характеризующее степень привилегированности того или иного класса трафика при использовании того или иного сетевого ресурса. Чем выше приоритет, тем меньше времени будут проводить пакеты данного класса в очередях к данному ресурсу.

Классификация трафика представляет собой отдельную задачу. Пакеты могут разбиваться на приоритетные классы на основании различных признаков: адрес назначения, адрес источника, идентификатор приложения, генерирующего этот трафик, а равно и любые другие комбинации признаков, которые содержатся в заголовках пакетов. Правила классификации пакетов отражают *политику администрирования* сети. Средства, выполняющие классификацию, — **классификаторы** — могут быть частью сетевых устройств. Масштабируемое решение — размещение механизмов классификации только в устройствах, расположенных на границе сети (например, в коммутаторах корпоративной сети, к которым подключаются компьютеры пользователей, или во входных маршрутизаторах сети поставщика услуг). Приоритеты могут также назначаться приложением на узле-отправителе.

Назначенный приоритет заносится в специальное поле заголовка пакета, после чего им могут воспользоваться остальные сетевые устройства, обрабатывающие трафик после классифицирующего устройства. Если в сети отсутствует централизованная политика назначения приоритетов, то каждое сетевое устройство может не согласиться с приоритетом, назначенным данному пакету в другой точке сети. В этом случае оно перепишет значение приоритета в соответствии со своей локальной политикой.

В сетевом устройстве, поддерживающем приоритетное обслуживание, имеется *несколько* очередей — по одной для каждого приоритетного класса. Пакет, поступивший в период перегрузки, помещается в очередь, соответствующую его приоритетному классу[1]. На рис. 5.14 приведен пример использования четырех приоритетных очередей с высоким, средним, нормальным и низким приоритетами. До тех пор пока из более приоритетной очереди не будут выбраны все имеющиеся в ней пакеты, устройство не переходит к обработке следующей, менее приоритетной очереди. Поэтому пакеты с низким приоритетом обрабатываются только тогда, когда пустеют все вышестоящие очереди.

---

[1] Иногда несколько очередей изображают в виде одной очереди, в которой находятся заявки различных классов. Если заявки выбираются из очереди в соответствии с их приоритетами, то это просто другое представление одного и того же механизма.

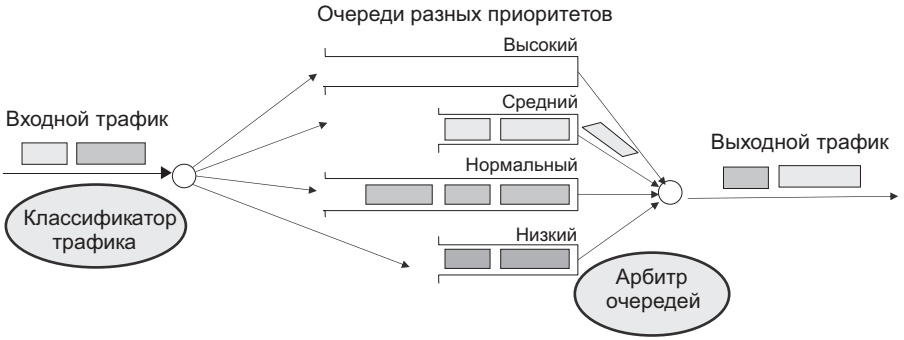


Рис. 5.14. Приоритетные очереди

Обычно по умолчанию всем приоритетным очередям отводятся буферы одинакового размера, но многие устройства разрешают администратору назначать каждой очереди буфер индивидуального размера. В идеальном случае размер буфера определяется таким, чтобы его хватало с некоторым запасом для хранения очереди среднестатистической длины. Однако поскольку определить это значение сложно, администраторы сети часто руководствуются следующим соображением: чем выше значимость трафика для предприятия, чем больше его интенсивность и пульсации, тем больший размер буфера требуется этому трафику. В примере, приведенном на рис. 5.14, для трафика высшего и нормального приоритетов выбраны большие размеры буферов, а для остальных двух классов — меньшие. Мотивы принятого решения для высшего приоритета очевидны, а для трафика нормального приоритета, возможно, были приняты во внимание высокая интенсивность и значительный коэффициент пульсаций.

Приоритетное обслуживание очередей обеспечивает высокое качество обслуживания для пакетов из самой приоритетной очереди. Если средняя интенсивность их поступления не превосходит пропускной способности устройства, то пакеты высшего приоритета всегда получают ту пропускную способность, которая им нужна. Время ожидания высокоприоритетных пакетов в очереди также минимально. Однако оно *не нулевое* — чем выше пульсации потока и его интенсивность, тем вероятнее возникновение очереди, образованной «своими» же высокоприоритетными пакетами. Трафик всех остальных приоритетных классов *почти прозрачен* для пакетов высшего приоритета. Слово «почти» относится к ситуации, когда высокоприоритетный пакет вынужден ждать завершения обслуживания низкоприоритетного пакета.

Что же касается остальных приоритетных классов, то качество их обслуживания будет ниже, чем у пакетов самого высокого приоритета, причем уровень снижения может быть очень существенным. Если коэффициент нагрузки выходного интерфейса, определяемый только трафиком высшего приоритетного класса, приближается в какой-то период времени к единице, то трафик остальных классов на это время просто замораживается. Поэтому приоритетное обслуживание обычно применяется для чувствительного к задержкам класса трафика, имеющего *небольшую интенсивность*. При таких условиях обслуживание этого класса не слишком ущемляет обслуживание остального трафика. Например, голосовой трафик чувствителен к задержкам, но его интенсивность обычно не превышает 8–16 Кбит/с, так что при назначении ему высшего приоритета ущерб остальным классам трафика оказывается не очень значительным.

Взвешенные очереди

Механизм взвешенных очередей разработан для того, чтобы можно было предоставить всем классам трафика определенный минимум пропускной способности. Под **весом** данного класса понимается доля предоставляемой классу трафика пропускной способности от полной пропускной способности выходного интерфейса.

При взвешенном обслуживании так же, как и при приоритетном, трафик делится на несколько классов, для каждого из которых ведется отдельная очередь пакетов. Но с каждой очередью связывается *не приоритет, а доля пропускной способности* ресурса, гарантируемая данному классу трафика при перегрузках этого ресурса. Для входного потока коммутатора таким ресурсом является коммутационный блок, а для выходного (после выполнения коммутации) — выходной интерфейс.

Поясним данный алгоритм управления очередями на примере. Показанное на рис. 5.15 устройство поддерживает для пяти классов трафика пять очередей к выходному интерфейсу коммутатора. Этим очередям при перегрузках выделяется соответственно 10, 10, 30, 20 и 30 % пропускной способности выходного интерфейса.

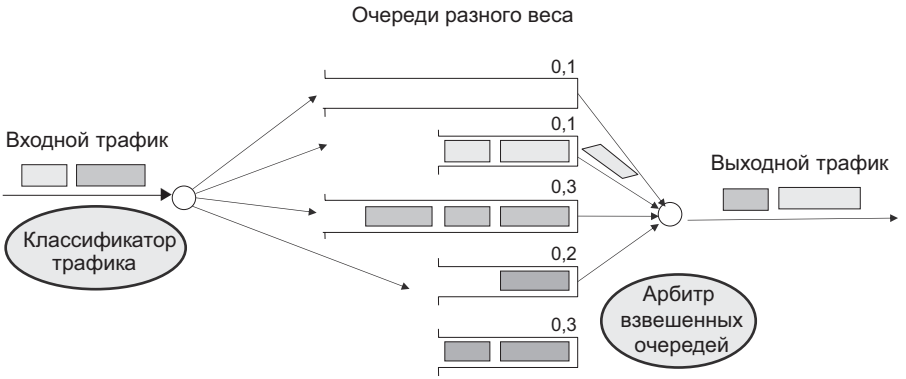


Рис. 5.15. Взвешенные очереди

Достигается поставленная цель за счет того, что очереди обслуживаются последовательно и циклически, причем в каждом цикле обслуживания из каждой очереди выбирается такое число байтов, которое соответствует весу данной очереди. Так, если цикл просмотра очередей в рассматриваемом примере равен одной секунде, а скорость выходного интерфейса составляет 1000 Мбит/с, то при перегрузках в каждом цикле первой очереди уделяется 10 % времени, то есть 0,1 секунды, в течение которой выбирается 100 Мбит данных. Аналогично, из второй — тоже 100 Мбит, из третьей — 300 Мбит, из четвертой — 200 Мбит, из пятой — 300 Мбит.

> В результате каждому классу трафика достается *гарантированный минимум пропускной способности,* что во многих случаях является более желательным результатом, чем подавление низкоприоритетных классов высокоприоритетными.

Так как данные выбираются из очереди пакетами, а не битами, то реальное распределение пропускной способности между классами трафика всегда немного отличается от плани-

руемого. Так, в предыдущем примере вместо 10 % первый класс трафика мог бы получать при перегрузках 9 или 12 %. Чем больше время цикла, тем точнее соблюдаются требуемые пропорции между классами трафика, так как из каждой очереди выбирается большее число пакетов, при этом влияние размера каждого пакета усредняется.

В то же время длительный цикл приводит к большим задержкам передачи пакетов. Так, при выбранном нами для примера цикле в одну секунду задержка может составить одну и более секунд — ведь арбитр возвращается к каждой очереди не чаще, чем раз в секунду, кроме того, за один раз из очереди могут быть выбраны не все пакеты. Поэтому при выборе времени цикла нужно обеспечить баланс между точностью соблюдения пропорций пропускной способности и стремлением к снижению задержки.

Для нашего примера более сбалансированным выглядит время цикла в 1^{-6} секунды, или 1000 мкс. С одной стороны, такое время гарантирует более низкий уровень задержек, так как очереди просматриваются намного чаще, чем при секундном цикле. С другой стороны, этого времени достаточно, чтобы выбрать из каждой очереди в среднем по несколько пакетов (первой очереди в нашем примере будет отводиться 100 мкс, что достаточно, например, для передачи в выходной канал десяти пакетов Gigabit Ethernet).

На уровень задержек пакетов, их вариации для некоторого класса трафика при взвешенном обслуживании в значительной степени влияет **относительный коэффициент загрузки**. В этом случае коэффициент подсчитывается как отношение интенсивности входного трафика некоторого класса к пропускной способности, выделенной этому классу в соответствии с его весом. Например, если в интерфейсе с пропускной способностью 1000 Мбит/с для некоторого потока выделено 10 %, то есть 100 Мбит/с, а средняя интенсивность потока, который попадает в эту очередь, равна 30 Мбит/с, то относительный коэффициент загрузки для этого потока составит 30/100 = 0,3. Качественное поведение очереди и, соответственно, задержек здесь выглядит примерно так же, как и для очереди FIFO, — чем меньше коэффициент загрузки, тем меньше средняя длина очереди и тем меньше задержки.

Еще одним вариантом взвешенного обслуживания является **взвешенное справедливое обслуживание** (Weighted Fair Queuing, WFQ). В случае подобного обслуживания пропускная способность ресурса делится между всеми потоками поровну, то есть «справедливо».

Взвешенное обслуживание обеспечивает требуемые соотношения между интенсивностями трафика различных очередей только *в периоды перегрузок*, когда каждая очередь постоянно заполнена. Если же какая-нибудь из очередей пуста (то есть для трафика данного класса текущий период не является периодом перегрузки), то при просмотре очередей она игнорируется, а ее время обслуживания распределяется между остальными очередями в соответствии с их весом.

Комбинированные алгоритмы обслуживания очередей

Каждый из описанных подходов имеет свои достоинства и недостатки. Приоритетное обслуживание, обеспечивая минимальный уровень задержек для очереди наивысшего приоритета, не дает никаких гарантий в отношении средней пропускной способности для трафика очередей более низких приоритетов. Взвешенное обслуживание обеспечивает заданное распределение средней пропускной способности, но не гарантирует выполнение требований к задержкам.

Существуют **комбинированные алгоритмы обслуживания очередей**. Например, в одном из алгоритмов подобного рода поддерживается одна приоритетная очередь, а остальные

очереди обслуживаются в соответствии со взвешенным алгоритмом. Обычно приоритетная очередь используется для чувствительного к задержкам трафика, а остальные — для эластичного трафика нескольких классов. Каждый класс эластичного трафика получает некоторый минимум пропускной способности при перегрузках. Этот минимум вычисляется как процент от пропускной способности, оставшейся от приоритетного трафика. Очевидно, что нужно как-то ограничить приоритетный трафик, чтобы он не поглощал всю пропускную способность ресурса. Обычно для этого применяются механизмы кондиционирования трафика, которые рассматриваются далее.

Механизмы кондиционирования трафика

Как мы помним, основной идеей методов QoS является выделение определенной доли пропускной способности определенным потокам трафика, при этом величина полученной потоком доли должна быть достаточной для того, чтобы качество обслуживания потока было удовлетворительным. Очереди с различными алгоритмами обслуживания позволяют реализовать только одну часть этой идеи — они выделяют определенную долю пропускной способности некоторому потоку пакетов. Однако если интенсивность входного потока по какой-то причине в некоторые периоды времени будет превышать выделенную ему пропускную способность, например, в виде пульсаций, то требуемые параметры качества обслуживания этого потока (уровни задержки, доля потерянных пакетов и др.) не будут обеспечены.

Во избежание этого рекомендуется применять механизмы **кондиционирования трафика**, включающие:

- классификацию;
- профилирование;
- сглаживание трафика.

Мы уже имели дело с **классификацией трафика** при изучении приоритетных и взвешенных очередей, когда предполагали наличие некоего механизма, решающего, какие пакеты нужно отправить в ту или иную очередь. В случае кондиционирования механизмы классификации потоков решают более общую задачу — они выполняют динамический анализ проходящих в сети потоков с целью отнесения их к тому или иному классу, каждый из которых должен обслуживаться сетью специфическим образом. К примеру, один класс потоков должен обслуживаться так, чтобы минимизировать потери пакетов, другой — минимизировать задержку, третий — вариацию задержки, а четвертому нужна гарантия пропускной способности. Для подобной классификации используются различные признаки пакетов, например адреса назначения и источника, тип протокола транспортного или прикладного уровня. Для каждого из классов в общем случае применяются разные методы QoS.

Профилирование (policing) представляет собой меру принудительного воздействия на трафик, направленного на ограничение скорости потока пакетов. Цель профилирования — добиться соответствия потока пакетов заданному скоростному **профилю** — набору заданных параметров потока. В качестве основного параметра обычно выступает средняя скорость потока пакетов, измеренная на определенном интервале времени[1]. Профилированный поток должен хорошо «укладываться» в выделенную для него пропускную способность.

[1] Применяются и более сложные варианты профилирования, например, учитывающие среднюю и пиковые скорости на нескольких временных отрезках.

При условии, что физическое оборудование сети остается неизменным, скорость потока пакетов можно уменьшить лишь двумя способами: либо заставить приложение-источник уменьшить интенсивность генерации пакетов (*подавление источника*), либо «проредить» поток. Последнюю операцию можно выполнить двумя путями: либо *отбрасывать* некоторые пакеты из потока, либо *задерживать* их на некоторое время в буфере. Отбрасывание и буферирование пакетов составляют суть методов профилирования и сглаживания трафика соответственно, которые мы и рассматриваем здесь (о подавлении источника см. в разделе «Обратная связь для предотвращения перегрузок»).

Рисунок 5.16 иллюстрирует действие механизма профилирования, показывая значения скорости трафика, измеренные на достаточно малых интервалах времени до и после профилирования. Как видно из рисунка, исходный поток имеет много пиков, которые выходят за границу определенной для данного потока пропускной способности. При профилировании часть пакетов, формирующих эти пульсации, отбрасывается, что приводит к удержанию скорости потока на заданном уровне на интервалах пульсаций и к сохранению исходной скорости в остальные периоды.

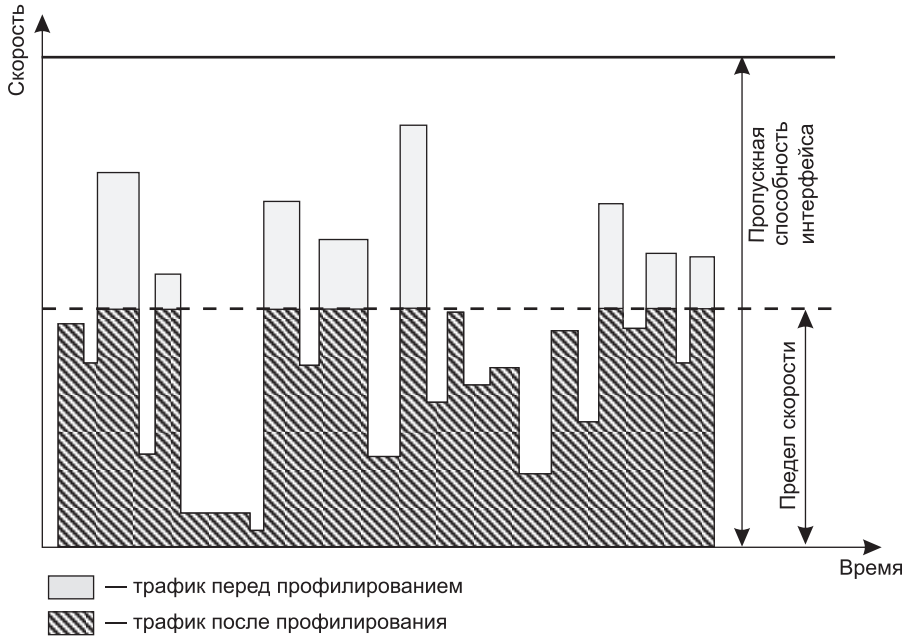


Рис. 5.16. Эффект профилирования — отбрасывание избыточного трафика

Трафик, кондиционированный таким образом, не будет нарушать расчетную, плановую загрузку сети. Однако для некоторых типов трафика отбрасывание пакетов может оказаться критичным.

Профилирование чаще всего применяют для ограничения трафика, поступающего в приоритетную очередь, так как только такими средствами можно предотвратить вытеснение всего остального трафика приоритетным трафиком.

Сглаживание трафика (shaping) в каком-то смысле подобно профилированию, так как оно имеет схожую цель — приведение параметров потока к заданному скоростному профилю.

Однако достигается эта цель другим способом. Вместо того чтобы отбрасывать избыточные пакеты, то есть те, передача которых могла бы привести к превышению лимита скорости, механизм сглаживания трафика *задерживает* пакеты-нарушители в специальном буфере так, что результирующая скорость потока оказывается в заданных пределах. Пакеты могут также *деквалифицироваться*, то есть перемещаться в класс обслуживания с более низкими привилегиями, например, переводятся из приоритетного класса в стандартный класс, обслуживаемый «по возможности».

Эффект сглаживания трафика иллюстрирует рис. 5.17. График скорости сглаживается за счет «срезания» выступов и заполнения впадин путем задержки пакетов, выходящих за уровень предельной скорости, и перемещения их в другие интервалы времени, в которых скорость оказывается меньше установленного предела. Как видим, формирование трафика является более «щадящим» механизмом кондиционирования — пакеты задерживают, но не отбрасывают.

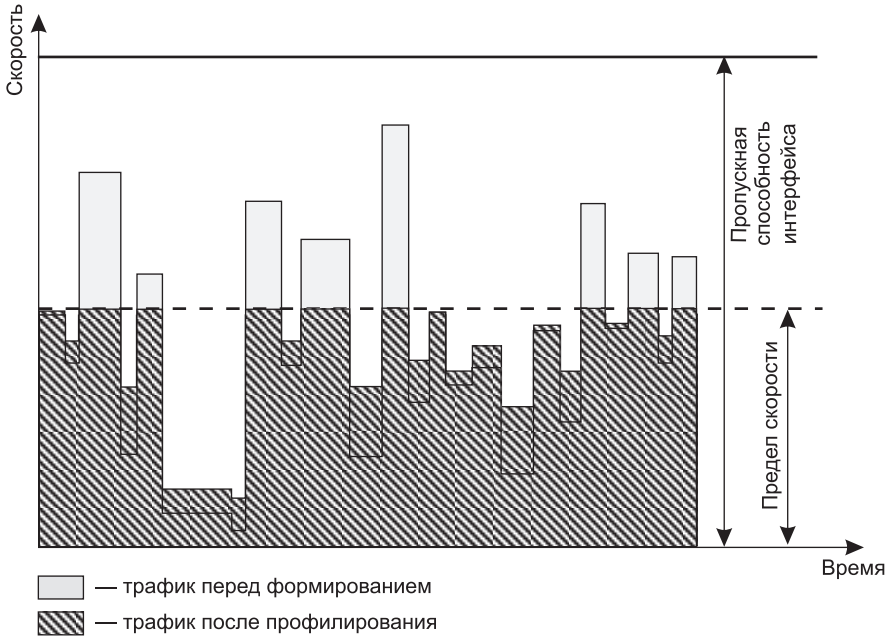


Рис. 5.17. Эффект сглаживания трафика

И профилирование и сглаживание устраняют скоростные выступы, *снижая пиковую скорость* кондиционируемого потока. Однако они по-разному воздействуют на его среднюю скорость. Очевидно, что из-за отбрасывания пакетов в результате *профилирования средняя скорость снижается.* А что происходит со скоростью во время сглаживания? Поскольку в этом случае пакеты, образующие пульсацию, задерживаются в буфере временно, а не исключаются из потока навсегда, то в результате *сглаживания средняя скорость не меняется.*

Обычно профилирование трафика выполняется маршрутизатором провайдера, принимающим трафик от сети пользователя. Это делается для защиты сети от пользовательского трафика, выходящего за пределы параметров, оговоренных в соглашении провайдера

и клиента. К тому же, если такое профилирование не делать, то от перегрузок сети пострадают все ее пользователи.

Сглаживание трафика считается делом самих пользователей, его должен выполнять пограничный маршрутизатор пользователя, посылающий данные в сеть провайдера. Если профиль, задаваемый для сглаживания трафика, совпадает с профилем последующего профилирования, то это гарантирует отсутствие потерь трафика из-за отбрасывания избыточных пакетов.

Известным алгоритмом, используемым как для сглаживания, так и для профилирования трафика, является **алгоритм ведра маркеров**. Его описание можно найти на сайте книги www.olifer.co.uk.

(S) *Алгоритм ведра маркеров*

Обратная связь для предотвращения перегрузок

Механизмы предотвращения перегрузок основаны на использовании *обратной связи*, с помощью которой перегруженный узел сети, реагируя на перегрузку, просит предыдущие узлы, расположенные вдоль маршрута следования потока (или потоков, принадлежащих к одному классу), *временно* снизить скорость трафика. После того как перегрузка в данном узле исчезнет, он посылает сообщение, разрешающее повысить скорость передачи данных.

Существуют несколько видов обратной связи (рис. 5.18). Они отличаются информацией, которая передается по обратной связи, а также тем, какой тип узла генерирует эту информацию и кто реагирует на эту информацию — конечный узел (компьютер) или промежуточный (коммутатор или маршрутизатор).

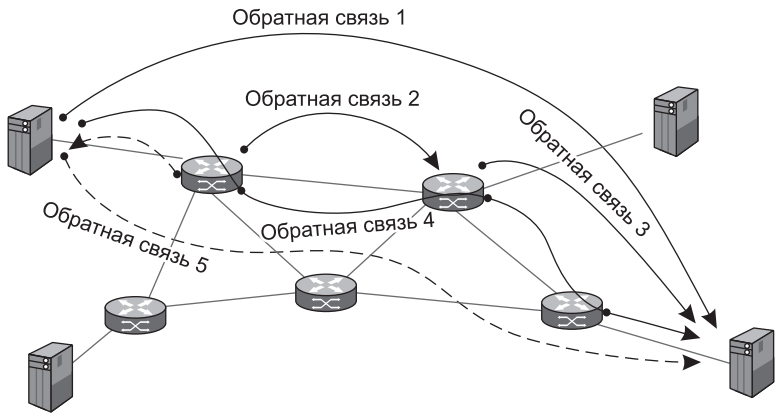


Рис. 5.18. Участники обратной связи

Обратная связь 1 *организована между двумя конечными узлами сети.* Сообщение о перегрузке порождается узлом-получателем и передается узлу-источнику. Этот вариант, имеющий специальное название — **управление потоком (flow control)**, обеспечивает наиболее

радикальное снижение нагрузки на сеть, так как только узел-источник может снизить скорость поступления информации в сеть. Целью этого вида обратной связи является борьба с перегрузками узла назначения, а не с перегрузками промежуточных сетевых устройств.[1] Устройства сети не принимают участия в работе этого вида механизма обратной связи, они только передают соответствующие *служебные сообщения* между конечными узлами. Время передачи этих сообщений по сети влияет на эффективность обратной связи.

Так, в высокоскоростных глобальных сетях за время, которое тратится на передачу сообщения о перегрузке узла назначения, узел-источник может успеть направить в сеть тысячи пакетов, из-за чего перегрузка не будет ликвидирована вовремя. Из теории автоматического управления известно, что задержки в контуре обратной связи могут приводить ко многим нежелательным эффектам, прямо противоположным первоначальным целям. Например, в системе могут начаться колебательные процессы, и она никогда не сможет прийти в равновесное состояние. Подобные явления наблюдались на ранней стадии развития интернета, когда из-за несовершенства алгоритмов обратной связи и маршрутизации в нем возникали участки перегрузок, которые периодически перемещались по сети. Причина такой проблемы интуитивно понятна — задержка в контуре обратной связи приводит к тому, что регулирующий элемент получает устаревшую информацию о состоянии регулируемого элемента. Поэтому возможны ситуации, когда узел-источник начинает снижать скорость передачи информации, хотя в действительности в узле-получателе уже нет очередей, и наоборот, повышать скорость передачи информации в тот момент, когда узел-получатель начал испытывать перегрузку. Для борьбы с такими явлениями в контур обратной связи обычно вводится интегрирующий элемент, который на каждом шаге обрабатывает не только текущее сообщение обратной связи, но и несколько предыдущих сообщений, что позволяет учесть динамику изменения ситуации и реагировать адекватно.

Обратная связь 2 *организована между двумя соседними коммутаторами.* Коммутатор сообщает соседу, находящемуся выше по течению потока, что он испытывает перегрузку и его буфер заполнился до критической величины. Получив такое сообщение, сосед, расположенный выше по течению, должен снизить на некоторое время скорость передачи данных в направлении перегруженного коммутатора и тем самым решить проблему перегрузки. Это менее эффективное для сети в целом решение, так как поток будет продолжать течь от узла-источника с той же скоростью, что и раньше. Однако для коммутатора, который испытывает перегрузку, это является хорошим выходом, так как он получает время, чтобы разгрузить переполнившуюся очередь. При этом проблема переносится в коммутатор, расположенный выше по течению, в котором теперь может возникнуть перегрузка, так как он начинает передавать данные из своего буфера с меньшей скоростью. Достоинством описанного метода является снижение задержки обратной связи, так как узлы являются соседями.

Обратная связь 3 *организована между некоторым промежуточным коммутатором и узлом-источником*; коммутатор посылает узлу-источнику сообщение о том, что он испытывает перегрузку, и просит его снизить интенсивность выходящего трафика. Все остальные промежуточные коммутаторы, лежащие между этими двумя узлами, только передают сообщения обратной связи в направлении к узлу-источнику, никак на них не реагируя.

В *обратной связи* 4, как и в обратной связи 1, сообщение о перегрузке генерируется узлом-получателем и передается узлу-источнику. Однако в данном случае каждый промежуточ-

---

[1] Один из самых известных алгоритмов управления потоком реализован в транспортном протоколе TCP, подробно описанном в главе 15.

ный коммутатор реагирует на это сообщение. Во-первых, он снижает скорость передачи данных в направлении узла назначения, во-вторых, он может изменить содержание сообщения. Например, если узел назначения просит снизить скорость до 300 Мбит/с, то промежуточный коммутатор может снизить эту величину до 200 Мбит/с, оценив состояние своего буфера. Кроме того, сгенерировать сообщение обратной связи может любой коммутатор сети, а не только узел назначения.

При описании различных вариантов организации обратной связи мы подразумевали, что сообщение о перегрузке идет в направлении, обратном направлению передачи пользовательской информации (отсюда и название механизма). Однако некоторые коммуникационные протоколы не предусматривают возможности генерации подобных сообщений промежуточными узлами. В таких условиях часто применяют искусственный прием — передачу сообщения о перегрузке узлу назначения, который преобразует его в сообщение обратной связи и отправляет в нужном направлении, то есть в направлении источника.[1] Этот вариант показан на рисунке как *обратная связь* 5.

В применяемых сегодня методах обратной связи используются следующие основные типы сообщений о перегрузке:

- признак перегрузки;
- максимальная разрешенная скорость передачи;
- максимальный разрешенный объем данных;
- косвенные признаки перегрузки.

Признак перегрузки не говорит о степени перегруженности сети или узла, а только фиксирует факт наличия перегрузки. Реакция узла, получившего такое сообщение, может быть разной. В некоторых протоколах узел обязан прекратить передачу информации в определенном направлении, до тех пор, пока не будет получено другое сообщение обратной связи, разрешающее продолжение передачи. В других протоколах узел ведет себя адаптивно, снижая скорость на некоторую величину и ожидая реакции сети. Если сообщения с признаком перегрузки продолжают поступать, то он продолжает снижение скорости.

Во втором типе сообщений указывается **максимальная скорость передачи**, то есть порог скорости, который должен соблюдать источник или промежуточный узел, расположенный выше по течению потока. В этом случае обязательно нужно учитывать время передачи сообщения по сети, чтобы исключить колебательные процессы в сети и обеспечить нужную скорость реакции на перегрузку. Поэтому в территориальных сетях такой способ обычно реализуется силами всех коммутаторов сети (обратная связь 4 в нашем примере).

Сообщение о **максимальном объеме данных** используется в широко применяемом в пакетных сетях алгоритме скользящего окна (подробнее о нем рассказывается в главе 15). Этот алгоритм позволяет не только обеспечивать надежную передачу данных, но и реализовать обратную связь для контроля потока между конечными узлами. Параметром, несущим информацию обратной связи, является «окно» — число, тесно связанное с текущим размером свободного пространства в буфере принимающего узла. Передающему узлу разрешено с любой скоростью передать объем информации, равный определенному для него окну. Но если этот лимит исчерпан, то передающий узел не имеет права передавать информацию, пока не получит следующее окно. При перегрузках принимающий узел уменьшает размер

[1] Подробнее об этом можно прочитать в разделе «Явное уведомление о перегрузке» главы 15.

окна, тем самым снижая нагрузку. Если эффект перегрузки исчезает, то принимающий узел увеличивает размер окна. Недостатком этого алгоритма является то, что он работает только в протоколах с установлением соединения.

В некоторых случаях передающий узел определяет, что принимающий узел (или узлы) испытывает перегрузку, по некоторым **косвенным признакам**, без получения сообщения обратной связи. Такими косвенными признаками могут быть факты потери пакетов. Примером протокола, использующего неявную информацию о перегрузках, является протокол TCP.

Резервирование ресурсов

Рассмотренные методы поддержания качества обслуживания ориентированы в основном на борьбу с перегрузками или предотвращение их в пределах отдельного узла сети. Вместе с тем понятно, что для поддержания гарантированного уровня качества обслуживания некоторого потока пакетов необходимо *скоординированное* применение этих методов на всем пути следования потока через сеть.

Резервирование ресурсов - это координирующая процедура, которая настраивает механизмы поддержания качества обслуживания вдоль следования потока таким образом, чтобы поток с некоторыми заданными характеристиками скорости был обслужен с заданными характеристиками QoS - задержками, потерями пакетов и др.

Основная идея процедуры резервирования ресурсов состоит в следующем. *Перед тем как* реальный поток данных будет направлен в сеть, каждому узлу сети вдоль маршрута его следования задается вопрос, может ли этот узел обслужить некоторый новый поток с заданными характеристиками QoS, если известны предельные характеристики скорости потока: средняя и пиковая скорости? Каждый узел при ответе на этот вопрос должен оценить свои возможности, то есть проверить, достаточно ли у него свободных ресурсов, чтобы принять на обслуживание новый поток и обслуживать его качественно. При положительном ответе узел должен некоторым образом зарезервировать часть своих ресурсов для данного потока, чтобы при поступлении пакетов потока на входные интерфейсы использовать эти ресурсы для обслуживания поступающих пакетов с гарантированным уровнем качества.

В общем случае каждый узел самостоятельно решает, какие ресурсы он должен зарезервировать для обслуживания некоторого потока с заданным качеством. Как показывает практика, основным ресурсом, требуемым для качественного обслуживания пакетов, является *пропускная способность интерфейса*, через который пакеты потока покидают узел. Поэтому в дальнейшем мы будем, несколько упрощая действительное положение дел, употреблять формулировку «резервирование пропускной способности» вместо «резервирование ресурсов».

Однако что же означает резервирование пропускной способности в сетях с коммутацией пакетов? Мы сталкивались с этой концепцией только при рассмотрении принципов функционирования сетей с коммутацией каналов. Действительно, для сетей с коммутацией пакетов механизм резервирования пропускной способности не является принципиально необходимым, он имеет *вспомогательное* значение и используется только в тех случаях, когда требуется обеспечение заданного качества обслуживания пакетов.

Процедура резервирования здесь подобна аналогичной процедуре в сетях с коммутацией каналов: определенному потоку данных назначается определенная часть пропускной

способности линии связи. Однако в сетях с коммутацией пакетов эта процедура является более гибкой, то есть если отведенная пропускная способность в какой-то период времени недоиспользуется потоком, то она может быть передана другим потокам. Еще одним отличием резервирования в пакетных сетях является то обстоятельство, что резервирование может выполняться не только «из конца в конец», но и для каких-то отдельных узлов по маршруту потока.

Процедура резервирования пропускной способности

Резервирование пропускной способности в пакетной сети «из конца в конец» начинается с операции, называемой контролем допуска в сеть потока, который просит зарезервировать для своего обслуживания некоторую пропускную способность сети между ее двумя конечными узлами.

Контроль допуска в сеть (admission control) состоит в проверке наличия доступной, то есть незарезервированной для других потоков пропускной способности на каждом из узлов сети на протяжении всего маршрута следования потока.

Контроль допуска может выполняться с помощью специального протокола, который передает по сети сообщение о запросе необходимой пропускной способности вдоль маршрута, по которому затем должны будут передаваться данные потока. Каждое коммуникационное устройство, приняв такое сообщение, должно решить, достаточно ли у него свободной (но еще не зарезервированной) пропускной способности, чтобы выделить новому потоку запрашиваемую величину. Отметим, что выполнение этой процедуры возможно и без такого протокола, если внешняя централизованная система или же администратор сети ведут учет и принимают решение о выделяемой пропускной способности.

Очевидно, что средняя скорость потока должна быть меньше, чем запрашиваемая пропускная способность, иначе поток будет обслужен с очень плохим качеством, даже несмотря на то, что ему была зарезервирована некоторая пропускная способность. В некоторых случаях при резервировании учитываются и пиковые скорости потока.

Если результат контроля допуска положителен в каждом узле (рис. 5.19), то сетевые устройства запоминают факт резервирования, чтобы при появлении пакетов данного потока распознать их и выделить им зарезервированную пропускную способность. Кроме того, при успешном резервировании доступная для резервирования (в будущем) пропускная способность уменьшается на величину, зарезервированную за данным потоком.

Теперь посмотрим, каким образом выполняется выделение пропускной способности потоку в моменты времени, когда его пакеты поступают на вход коммуникационного устройства $S2$, которое запомнило факт резервирования пропускной способности для потока $F1$ на выходном интерфейсе $P2$ (рис. 5.20).

Такое выделение пропускной способности можно обеспечить разными способами, в том числе и с использованием взвешенных очередей. Пусть потоку $F1$ при резервировании было выделено 25 % пропускной способности интерфейса $P2$. Будем считать, что резервирование было выполнено только для потока $F1$, а для других потоков, которые проходят через выходной интерфейс $P2$, резервирование не производилось.

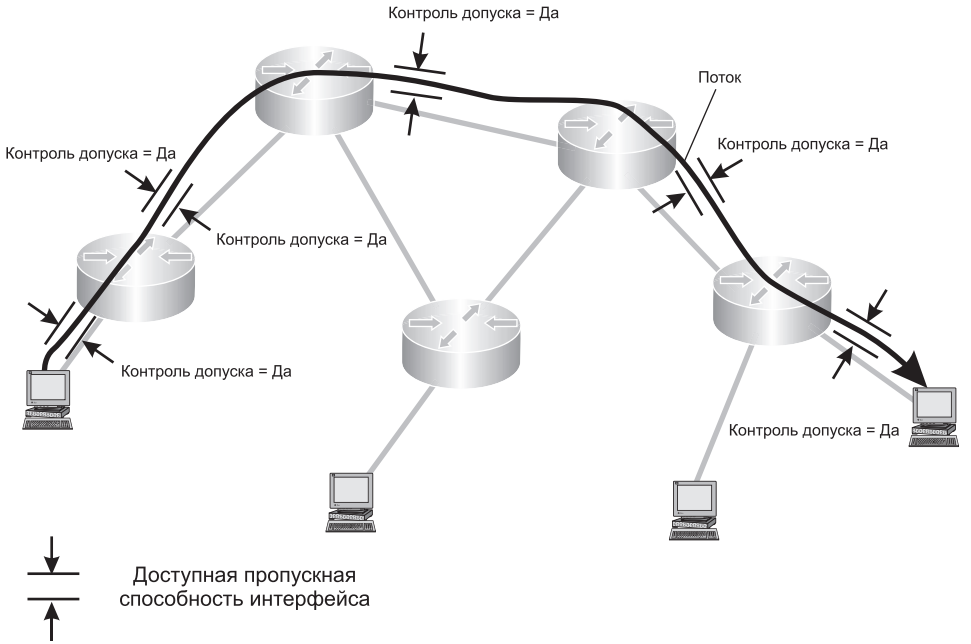


Рис. 5.19. Контроль допуска потока

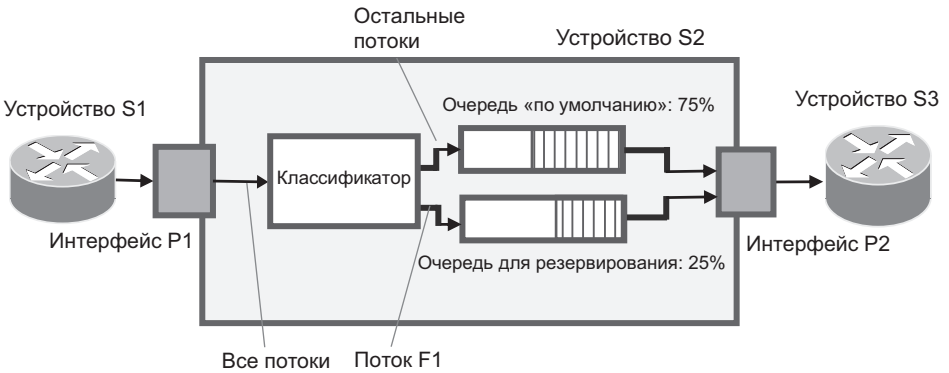


Рис. 5.20. Выделение зарезервированной пропускной способности

Чтобы добиться желаемого результата, достаточно организовать для выходного интерфейса *две взвешенные очереди* — очередь для потока *F*1 с весом 25 % и очередь «по умолчанию» для всех остальных потоков. Кроме того, необходимо активизировать *классификатор*, который будет проверять пакеты на всех входных интерфейсах устройства *S*2 (на рис. 6.14 показан только один входной интерфейс *P*1), отбирать пакеты потока *F*1 по заданным при резервировании признакам и направлять их в очередь для потока *F*1. В те периоды времени, когда скорость потока *F*1 окажется меньше зарезервированной пропускной способности в 25 %, неиспользованная ее часть будет потребляться потоками из очереди «по умолчанию» — в силу алгоритма работы взвешенных очередей. Зато в периоды, когда скорость

потока *F*1 достигнет заявленного максимума потребления пропускной способности в 25 %, все остальные потоки будут довольствоваться оставшимися 75 %.

В описанном примере не задействован механизм *профилирования* трафика. При наличии отдельной взвешенной очереди для потока, зарезервировавшего пропускную способность, этот механизм не является обязательным, так как сам механизм взвешенных очередей ограничит пропускную способность потока в нужных пределах в периоды перегрузок, когда все взвешенные очереди заполняются полностью.

Для той же цели можно задействовать *приоритетные очереди*. Применение приоритетной очереди может оказаться необходимым, если потоку, помимо определенного уровня пропускной способности, требуется обеспечить и минимально возможный уровень задержек пакетов. При использовании приоритетной очереди профилирование необходимо всегда, так как приоритетный механизм не обеспечивает ограничения скорости потока, что может позволить приоритетному потоку захватить весь ресурс.

Нужно подчеркнуть, что резервирование приводит к ожидаемым результатам только в тех случаях, когда реальная средняя скорость потоков, для которых было выполнено резервирование, оказывается не выше, чем пропускная способность, запрошенная при резервировании и реализованная при конфигурировании сетевых устройств. В противном случае результаты могут оказаться даже хуже, чем при наличии единственной очереди «по умолчанию» и обслуживании «по возможности». Так, если скорость потока окажется выше, чем предел, учитываемый механизмом профилирования, то часть пакетов будет отброшена даже в том случае, если устройство не перегружено и могло бы отлично справиться с предложенным трафиком без применения механизмов QoS.

Обеспечение заданного уровня задержек

При описании процедуры резервирования пропускной способности мы сфокусировались на механизмах выделения пропускной способности некоторому потоку и оставили без внимания один важный вопрос: *какую пропускную способность должен запрашивать поток для того, чтобы задержки его пакетов не превышали некоторой величины?* Единственное соображение, которое было высказано по этому поводу, заключалось в том, что запрашиваемая пропускная способность должна быть выше, чем средняя скорость потока, иначе значительная часть пакетов просто может постоянно отбрасываться сетью, так что качество обслуживания окажется гарантированно низким.

Этот вопрос на самом деле оборачивается сложной проблемой, так как мы не можем, например, сконфигурировать очередь приоритетного или взвешенного обслуживания так, чтобы она строго обеспечила какой-либо заранее заданный порог задержек и их вариации. Направление пакетов в приоритетную очередь только позволяет гарантировать, что задержки будут достаточно низкими — существенно ниже, чем у пакетов, которые обрабатываются в очереди по умолчанию. Мы также знаем, что при наличии взвешенных очередей задержки будут снижаться со снижением относительного коэффициента использования пропускной способности, отведенной очереди. Все сказанное относится к качественным характеристикам задержек, а вот количественно оценить их значение очень сложно.

Каким же образом поставщик услуг может выполнить свои обязательства перед клиентами? Очень «просто» — он должен постоянно *измерять фактические значения характеристик трафика в сети* и гарантировать пользователям сети величины задержек в соответствии с наблюдаемыми результатами. На практике постоянный мониторинг задержек

и потерь пакетов требует установки в сети большого количества агентов-измерителей, хорошо синхронизированных друг с другом, а также программной системы регистрации и анализа измерительной информации.

Инжиниринг трафика

При рассмотрении системы обеспечения качества обслуживания, основанной на резервировании, мы не стали затрагивать вопрос маршрутов следования потоков через сеть. Точнее, мы исходили из того, что маршруты каким-то образом выбраны, причем этот выбор делается без учета требований QoS. И в условиях заданности маршрутов мы старались обеспечить прохождение по этим маршрутам такого набора потоков, для которого можно гарантировать соблюдение требований QoS.

Очевидно, что задачу обеспечения требований QoS можно решить более эффективно, если считать, что маршруты следования трафика не фиксированы, а также подлежат выбору. Это позволило бы сети обслуживать больше потоков с гарантиями QoS при тех же характеристиках самой сети, то есть пропускной способности каналов и производительности коммутаторов и маршрутизаторов.

Задачу выбора маршрутов для потоков (или классов трафика) с учетом требований QoS решают методы **инжиниринга трафика** (Traffic Engineering). С помощью этих методов стремятся по возможности сбалансированно загрузить все ресурсы сети, чтобы сеть при заданном уровне качества обслуживания обладала как можно более высокой *суммарной производительностью*.

Методы инжиниринга трафика не только позволяют найти рациональный маршрут для потока, но и резервируют для него ресурсы, главным образом пропускную способность устройств сети вдоль этого маршрута.

Недостатки традиционных методов маршрутизации

В сетях с коммутацией пакетов традиционные протоколы маршрутизации[1] осуществляют выбор маршрута на основе топологии сети без учета ее текущей загрузки. Для каждой пары «адрес источника — адрес назначения» такие протоколы выбирают единственный маршрут, не принимая во внимание информационные потоки, протекающие через сеть. В результате все потоки между парами конечных узлов сети идут по *кратчайшему* (в соответствии с некоторой метрикой) маршруту. Выбранный маршрут может быть более рациональным, например, если в расчет принимается номинальная пропускная способность каналов связи или вносимые ими задержки, или менее рациональным, если учитывается только количество промежуточных маршрутизаторов между исходным и конечным узлами.

Классическим примером неэффективности такого подхода является так называемая «рыба» — сеть с топологией, приведенной на рис. 5.21. Несмотря на то что между коммутаторами *A* и *E* существует два пути (верхний — через коммутатор *B*, нижний — через коммутаторы *C* и *D*), весь трафик от коммутатора *A* к коммутатору *E* в соответствии с тра-

[1] Подробно о протоколах маршрутизации читайте в главе 16.

дициональными принципами маршрутизации направляется по верхнему пути. Только потому, что нижний путь немного (на один ретрансляционный участок) длиннее, чем верхний, он игнорируется, хотя мог бы работать «параллельно» с верхним путем.

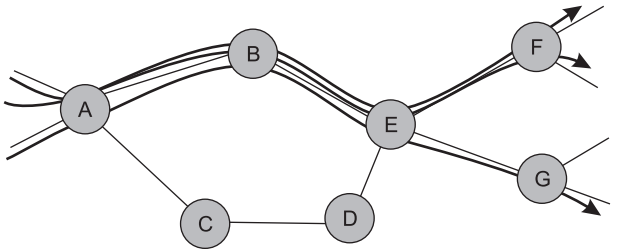


Рис. 5.21. Неэффективность кратчайших путей

Налицо явная ущербность разделения ресурсов сети между потоками — одни ресурсы работают с перегрузкой, а другие при этом не используются вовсе. Традиционные методы борьбы с перегрузками эту проблему решить не могут, нужны качественно иные механизмы.

Методы инжиниринга трафика

Исходными данными для методов инжиниринга трафика являются:

- характеристики передающей сети;
- сведения о предложенной нагрузке сети.

К *характеристикам передающей сети* относится ее топология, а также производительность составляющих ее коммутаторов и линий связи. Предполагается, что производительность процессора каждого коммутатора достаточна для обслуживания трафика всех его входных интерфейсов, даже если трафик поступает на интерфейс с максимально возможной скоростью, равной пропускной способности интерфейса. При таких условиях в качестве резервируемых ресурсов выступает пропускная способность линий связи между коммутаторами (рис. 5.22).

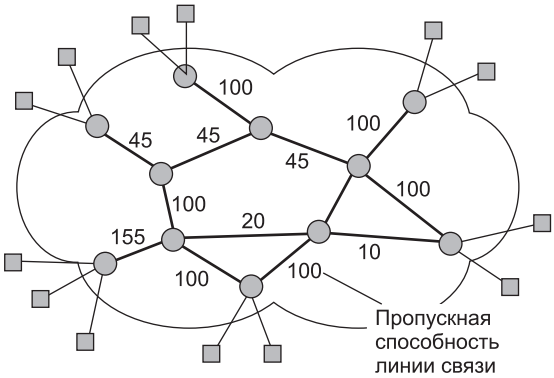


Рис. 5.22. Топология сети и производительность ее ресурсов

Сведения о предложенной нагрузке сети представляют собой информацию о потоках данных, которые сеть должна передать между своими пограничными коммутаторами. Каждый поток характеризуется точкой входа в сеть, точкой выхода из сети и профилем трафика. Для получения оптимальных решений желательно использовать детальное описание каждого потока, например, учитывать величину возможной пульсации трафика. Однако, как правило, учитываются только их средние скорости передачи данных (рис. 5.23).

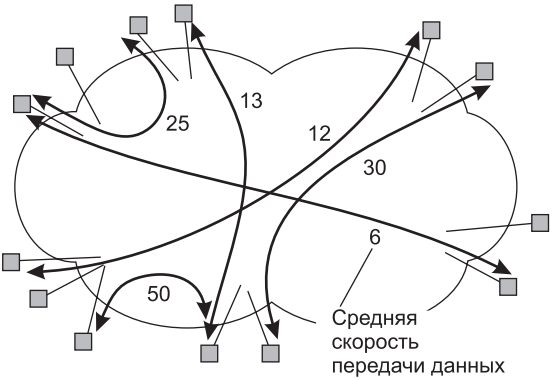


Рис. 5.23. Предложенная нагрузка

Методы инжиниринга трафика чаще применяют не к отдельным, а к *агрегированным* потокам, которые являются объединением нескольких потоков. Так как мы ищем общий маршрут для нескольких потоков, то агрегировать можно только потоки, имеющие общие точки входа в сеть и выхода из сети. Агрегированное задание потоков позволяет упростить задачу выбора путей, так как при индивидуальном рассмотрении каждого пользовательского потока промежуточные коммутаторы должны хранить слишком большие объемы информации. Агрегирование отдельных потоков в один возможно только в том случае, когда все потоки, составляющие агрегированный поток, *предъявляют одни и те же требования к качеству обслуживания*. Далее в этом разделе мы будем для краткости пользоваться термином «поток» как для индивидуального потока, так и для агрегированного, поскольку принципы инжиниринга трафика от этого не меняются.

> Решением задачи инжиниринга трафика является такой набор маршрутов для заданного множества потоков трафика, для которого все значения коэффициентов загрузки ресурсов вдоль маршрута следования каждого потока не превышают некоторого заданного порога Kmax.

Максимальный уровень использования ресурсов K_{max} выбирается таким образом, чтобы механизмы управления перегрузкой могли обеспечить требуемое качество обслуживания. Это означает, что для эластичного трафика максимальное значение выбирается не больше чем 0,9, а для чувствительного к задержкам трафика — не больше чем 0,5. Так как обычно резервирование производится не для всех потоков, нужно оставить часть пропускной способности для свободного использования. Поэтому приведенные максимальные значения обычно уменьшают до 0,75 и 0,25 соответственно. Для упрощения рассуждений будем полагать далее, что в сети передается один вид трафика, а далее покажем, как обобщить решение задачи инжиниринга для случая трафика нескольких типов.

Решение задачи инжиниринга трафика можно искать заблаговременно, *в фоновом режиме*. Для этих целей должна быть разработана внешняя по отношению к сети программа, исходными данными для которой служат топология и производительность сети, а также предложенная нагрузка. Эта программа ищет оптимальное распределение маршрутов, например, путем направленного перебора вариантов.

Задачу инжиниринга трафика можно решать *в оперативном режиме*, поручив ее самим коммутаторам сети. Для этого используются стандартные протоколы маршрутизации, модифицированные для инжиниринга трафика. Модификация протоколов маршрутизации состоит в том, что они сообщают друг другу не только топологическую информацию, но и текущее значение свободной пропускной способности для каждого ресурса.

После того как решение найдено, нужно его реализовать, то есть отразить в таблицах маршрутизации. Если мы вознамеримся проложить эти маршруты в дейтаграммной сети, то рискуем столкнуться с еще одной проблемой. Дело в том, что таблицы маршрутизации в них учитывают только адреса назначения пакетов. Коммутаторы и маршрутизаторы таких сетей (например, IP-сетей) не работают с потоками, поскольку для них поток в явном виде не существует, а каждый пакет при его продвижении является независимой единицей коммутации. В связи с этим методы инжиниринга трафика сегодня используются только в сетях с виртуальными каналами, для которых не составляет труда реализовать найденное решение для группы потоков.

Работа в недогруженном режиме

Как отмечалось, самым простым способом обеспечения требований QoS для всех потоков является работа сети в недогруженном режиме, или с избыточной пропускной способностью.

Говорят, что *сеть имеет избыточную пропускную способность*, когда все части сети в любой момент времени обладают такой пропускной способностью, которой достаточно, чтобы обслужить все потоки трафика, протекающего в это время через сеть, с удовлетворительными характеристиками производительности и надежности. Другими словами, ни одно из сетевых устройств такой сети никогда не подвергается перегрузкам, которые могли бы привести к значительным задержкам или потерям пакетов из-за переполнения очередей пакетов (конечно, это не исключает случаев потерь сетью пакетов по другим причинам, не связанным с перегрузкой сети, например, из-за искажений сигналов в линиях связи либо отказов сетевых узлов или линий связи).

Заметим, что приведенное определение сети с избыточной пропускной способностью намеренно упрощено, чтобы донести суть идеи. Более аккуратное определение должно было бы учитывать случайный характер протекающих в сети процессов и оперировать статистическими определениями событий. Например, оговаривать, что такие события, как длительные задержки или потери пакетов из-за переполнения очередей в сети с избыточной пропускной способностью, случаются настолько редко, что ими можно пренебречь.

Простота обеспечения требований QoS за счет работы сети в недогруженном режиме является главным достоинством этого подхода — он требует только увеличения пропускной способности линий связи и, соответственно, производительности коммуникационных устройств сети. Никаких дополнительных усилий по исследованию характеристик потоков сети и конфигурированию дополнительных очередей и механизмов кондиционирования трафика здесь не требуется.

Чтобы быть уверенными, что сеть обладает достаточной пропускной способностью для качественной передачи трафика, необходим постоянный мониторинг временны́х характеристик (задержек и их вариаций) процессов передачи пакетов сетью. А в том случае, когда результаты мониторинга начинают стабильно показывать ухудшение характеристик качества обслуживания, необходимо провести очередную модернизацию сети и увеличить пропускную способность линий связи и коммуникационных устройств.

Однако мониторинг задержек и их вариаций является трудоемкой работой. Обычно операторы, которые хотят поддерживать свою сеть в недогруженном состоянии и за счет этого обеспечивать высокое качество обслуживания, просто осуществляют мониторинг уровня трафика в линиях связи сети, то есть *измеряют коэффициент использования пропускной способности линий связи*. При этом линия связи считается недогруженной, если ее коэффициент использования постоянно не превосходит некоторый достаточно низкий уровень, например 20–30 %. Имея такие значения измерений, можно считать, что линия в среднем не испытывает перегрузок, а значит, сеть будет передавать трафик качественно.

Для автоматического резервирования ресурсов маршрутизаторов (таких, как пропускная способность интерфейсов, размеры буферов) в пределах, разрешенных политикой QoS для данной сети, в начале 90-х годов был разработан **протокол резервирования ресурсов (Resource reSerVation Protocol, RSVP)**. Этот протокол принимает запрос на резервирования ресурсов от приложения, работающего на узле сети, генерирующего некоторый поток пакетов. В запросе указываются параметры, рекомендуемые для качественной передачи сетью трафика этого приложения: верхнюю и нижнюю границы требуемой пропускной способности, максимальные значения задержки и ее вариации и т. п. Запрос передается в пакете протокола RSVP всем маршрутизаторам, расположенным *вдоль пути потока* между конечными узлами пути потока.

Каждый маршрутизатор, получив запрос RSVP, проверяет, имеются ли у него ресурсы, необходимые для поддержания запрашиваемой пропускной способности и уровня QoS. Если запрос принимается, то маршрутизатор посылает запрос далее, вдоль маршрута следующему маршрутизатору, а данные о требуемом уровне QoS передаются тем механизмам маршрутизатора, которые ответственны за управление трафиком. Если все маршрутизаторы вдоль пути потока принимают запрос RSVP, то в сети устанавливается состояние резервирования для данного потока, так что пакеты потока, генерируемого узлом-отправителем, получают необходимый уровень обслуживания.

Подробнее о протоколе RSVP можно прочитать на сайте книги www.olifer.co.uk.

(S) *Протоколы IntServ и Differv. Поддержка QoS маршрутизаторами.*

Вопросы к части I

1. Поясните, почему сети WAN появились раньше, чем сети LAN.
2. Охарактеризуйте интернет в соответствии с критериями классификации компьютерных сетей.
3. В чем главное отличие многотерминальной системы от компьютерной сети?
4. Дайте определения терминам «клиент», «сервер», «сетевая служба», «сетевой сервис», «сетевая услуга».
5. Какой тип топологии наиболее распространен сегодня в локальных сетях?
6. Укажите, какие параметры передаваемых данных могут служить признаком информационного потока:
   а) адрес назначения;
   б) адрес источника;
   в) тип приложения;
   г) номер интерфейса, на который поступил пакет.
7. Какие из перечисленных функций включены в задачу продвижения данных:
   а) распознавание потока;
   б) просмотр таблицы маршрутизации;
   в) внесение информации о маршруте в таблицу маршрутизации;
   г) передача данных на заданный интерфейс маршрутизатора.
8. Какие из перечисленных функций включены в задачу маршрутизации:
   а) выбор маршрутов;
   б) формирование таблиц маршрутизации;
   в) реализация доступа к разделяемой среде;
   г) обновление таблиц маршрутизации.
9. Какие из следующих утверждений всегда верны:
   а) скорость передачи данных может быть выше предложенной нагрузки (речь идет об одних и тех же данных);
   б) скорость передачи данных всегда ниже пропускной способности;
   в) пропускная способность никак не связана с предложенной нагрузкой;
   г) скорость передачи данных может быть ниже предложенной нагрузки.
10. Можно ли организовать надежную передачу данных между двумя конечными узлами без установления логического соединения?
11. В каких сетях прокладываются виртуальные каналы?
    а) в сетях с коммутацией каналов;
    б) в сетях с коммутаций пакетов;
    в) и в тех и в других.

12. Какое логическое соединение может быть названо виртуальным каналом?
13. Укажите, какие из приведенных ниже утверждений верны при любых условиях:
    а) в сетях с коммутацией каналов необходимо предварительно устанавливать соединение;
    б) в сетях с коммутацией каналов не требуется указывать адрес назначения данных;
    в) сеть с коммутацией пакетов более эффективна, чем сеть с коммутацией каналов;
    г) сеть с коммутацией каналов предоставляет взаимодействующим абонентам гарантированную пропускную способность;
    д) составной канал постоянно закрепляется за двумя абонентами;
    е) составной канал имеет постоянную и фиксированную пропускную способность на всем своем протяжении.
14. Какие свойства сетей с коммутацией каналов свидетельствуют об их недостатках?
15. Какие свойства сетей с коммутацией пакетов негативно сказываются на передаче мультимедийной информации?
16. Какие из следующих характеристик полностью определяются пропускной способностью канала:
    а) время коммутации пакета;
    б) время распространения сигнала;
    в) время ожидания пакета в очереди;
    г) время сериализации.
17. Приведите расчеты, поясняющие значение элементарного канала цифровых телефонных сетей 64 Кбит/с.
18. Пусть линия связи, подключающая абонента к телефонной сети, имеет пропускную способность 128 Кбит/с. Почему линия связи, соединяющая коммутаторы в этой сети, не может иметь пропускную способность 30 710 Кбит/с?
19. Представим себе писателя, который пишет книгу и по мере готовности каждой новой главы посылает ее в издательство. Работая над главой 8, он получил от издателя письмо о том, что им не получена глава 5. Писатель еще раз послал письмо с главой 5, попросив в дальнейшем подтверждать получение каждой из следующих глав. Какой способ продвижения пакетов — дейтаграммный или метод с установлением логического соединения — ближе к описанному методу обмена почтовыми отправлениями?
20. Из-за чего скорость передачи пользовательских данных в сетях с коммутацией пакетов всегда ниже пропускной способности каналов связи? Варианты ответов:
    а) из-за пульсаций трафика;
    б) из-за разделения линий связи с другими пользователями;
    в) из-за наличия заголовков у пакетов;
    г) из-за отсутствия логических соединений;
    д) из-за задержек на коммутаторах.
21. Поясните, почему пакетный коммутатор имеет буферную память:
    а) для принятия решения о коммутации пакета требуется выполнить анализ заголовка пакета;

б) коммутатор иногда не успевает обработать пакет до прихода следующего;

в) для ускорения обработки пакетов их помещают в буферную память;

г) скорость передачи данных в одной линии связи коммутатора выше, чем в другой.

22. В чем сходство и различие составного и виртуального каналов?
23. Пусть имеются пять различных методов передвижения объектов: доставка товаров по почте; движение автотранспорта; лесосплав; рассылка спама; движение городских автобусов. Укажите, какой из этих методов может служить аналогией для следующих методов продвижения в компьютерных сетях: дейтаграммная пакетная коммутация; коммутация каналов; пакетная коммутация на основе виртуальных каналов; пакетная коммутация на основе логических соединений.
24. Какой из способов продвижения данных — коммутация каналов или коммутация пакетов, — по вашему мнению, более защищен от информационных атак?
25. На каком уровне модели OSI работают сетевые службы? Приложения?
26. Можно ли представить другой вариант модели взаимодействия сетевых средств, отличный от модели OSI и имеющий, например, шесть уровней?
27. Всегда ли справедливо утверждение: «Протокол — это стандарт, формализованно описывающий правила взаимодействия двух систем, включая последовательность обмена сообщениями и их форматы»?
28. Пусть на двух компьютерах установлено идентичное программное и аппаратное обеспечение, за исключением того, что драйверы сетевых адаптеров Ethernet поддерживают разные интерфейсы с вышележащим протоколом сетевого уровня IP. Будут ли эти компьютеры нормально взаимодействовать, если их соединить в сеть?
29. Если протокольная единица данных (PDU) некоторого протокола инкапсулирована в PDU протокола k-го уровня модели OSI, можно ли с уверенностью утверждать, что первый протокол относится к уровню $k + 1$?
30. Какие преимущества дает следование открытым спецификациям?
31. Вычислите значение стандартного отклонения и коэффициента вариации случайной задержки по гистограмме на рис. 5.3.
32. Как можно уменьшить скорость потока пакетов?
33. Если скорость потока, передаваемого по каналу с пропускной способностью 100 Гбит/с, равна 50 Гбит/с, чему равна скорость передачи отдельного пакета из этого потока?
34. Пусть для передачи голоса используется дискретизация по времени с интервалом 25 мкс и дискретизация по значениям — 1024 градации звукового сигнала. Какая пропускная способность необходима для передачи полученного таким образом голосового трафика?
35. Какой тип обслуживания целесообразно применить, если нужно обеспечить различную минимальную гарантированную способность трем классам трафика?
36. Объясните причину возможного возникновения очередей даже при невысокой и средней загрузке коммутаторов или маршрутизаторов сети с коммутацией пакетов.
37. Укажите, какие из перечисленных приложений наиболее чувствительны к потерям пакетов, задержкам и/или вариациям задержек: файловый сервис; мультимедийные сервисы; электронная почта; текстовый редактор; программа управления роботом; широковещательная рассылка аудиоинформации.

38. Может ли трафик передаваться с большими задержками, но без джиттера?
39. К каким нежелательным последствиям может привести приоритетное обслуживание?
40. Пусть в результате измерений было получено среднее значение времени оборота пакетов некоторого типа. Какие причины могут вызвать погрешность этой величины, если использовалась активная схема измерений? А если, наоборот, пассивная?
41. Какой параметр трафика меняется при инжиниринге трафика?
42. Пусть в коммутатор поступает поток данных, средняя скорость которого равна 10 000 пакетов в секунду, и пусть среднее время обработки пакета в коммутационном блоке равно 1 мкс. Нужен ли буфер для очереди перед коммутационным блоком?
43. Пусть пропускная способность 1 Гбит/с выходного интерфейса коммутатора должна быть разделена между тремя классами трафика в отношении 1:4:15. Каким образом эта проблема может быть решена с использованием взвешенных очередей? Какой объем данных из каждой очереди должен выбираться при каждом просмотре, если цикл просмотра равен 100 мкс?
44. К какому типу приложений может быть отнесена электронная почта?
    а) интерактивное;
    б) асинхронное;
    в) чувствительное к задержкам;
    г) чувствительное к потере данных;
    д) устойчивое к потере данных.
45. Как влияют профилирование и сглаживание на пиковую и среднюю скорость кондиционируемых потоков?
46. Известно, что профилирование — один из механизмов повышения качества обслуживания. Известно также, что в процессе профилирования некоторые пакеты отбрасываются. Потери пакетов ухудшают качество обслуживания. Как объяснить это противоречие?
47. Верно ли утверждение: в системе обслуживания с абсолютными приоритетами время пребывания в очереди пакетов наивысшего приоритета равно нулю?

Часть II

Технологии физического уровня проводных сетей

- ❑ Глава 6. Линии связи
- ❑ Глава 7. Кодирование и мультиплексирование данных
- ❑ Глава 8. Технологии первичных сетей PDH и SDH
- ❑ Глава 9. Технологии первичных сетей DWDM и OTN

В главе 6 рассматриваются различные типы линий связи и их характеристики. Сигналы, переданные по линии связи, приходят к приемнику с задержкой и искажениями, вызванными различными причинами. Так, скорость передачи данных ограничивается пропускной способностью линии. Искажения сигналов возникают в результате затухания мощности сигнала, ограниченности полосы пропускания и внешних и внутренних помех. Изучение этих характеристик базируется на аппарате спектрального представления электрических и оптических сигналов. Пропускная способность линии зависит от ее полосы пропускания и метода кодирования передаваемой по линии связи информации.

Глава 7 посвящена методам кодирования аналоговой и дискретной информации. Эти методы различаются шириной спектра сигнала, полученного в результате кодирования, устойчивостью к шумам, способностью обнаруживать и исправлять битовые ошибки, некоторыми другими характеристиками. В этой же главе изучаются частотное, волновое и временное синхронное и асинхронное мультиплексирование каналов.

Главы 8 и 9 посвящены технологиям первичных сетей PDH, SDH, DWDM и OTN. После изучения принципов построения компьютерных сетей в воображении читателя могла возникнуть достаточно простая картина компьютерной сети — компьютеры и коммутаторы, соединенные друг с другом отрезками кабеля. Однако при более детальном рассмотрении компьютерной сети все оказывается сложнее, чем казалось при изучении модели OSI. Дело в том, что цельные отрезки кабеля используются для соединения сетевых устройств только на небольших расстояниях, то есть в локальных сетях. При построении сетей WAN и MAN такой подход крайне расточителен из-за высокой стоимости протяженных линий связи такого рода. Поэтому гораздо чаще для связи коммутаторов в сетях WAN и MAN используются существующие первичные территориальные сети с коммутацией каналов. В этом случае в первичной сети создается составной канал, который для компьютерной или телефонной наложенной сети выполняет те же функции, что и отрезок кабеля, — обеспечивает физическое двухточечное соединение. Технологии первичных сетей существенно отличаются от технологий компьютерных пакетных сетей: они основаны на принципе коммутации каналов; имеют узкую специализацию — транспортировку данных; используют в качестве способа разделения передающей среды методы синхронного временного мультиплексирования каналов (сети PDH, SDH и OTN), а также мультиплексирования световых волн различной длины (сети DWDM).

ГЛАВА 6 Линии связи

Классификация линий связи

Первичные сети, линии и каналы связи

При описании технической системы, которая передает информацию между узлами сети, в литературе можно встретить несколько названий: *линия связи*, *составной канал*, *канал*, *звено*. Часто эти термины используются как синонимы, причем во многих случаях это не вызывает проблем. Вместе с тем терминам присущи и смысловые различия.

- **Звено** (link) — это сегмент, обеспечивающий передачу данных между двумя соседними узлами сети. Звено не содержит промежуточных устройств коммутации и мультиплексирования, но может включать усилители и регенераторы сигналов.
- **Каналом** (channel) чаще всего обозначают часть пропускной способности звена, используемую независимо при коммутации. Например, звено первичной сети может состоять из 30 каналов, каждый из которых обладает пропускной способностью 64 Кбит/с.
- **Составной канал** (circuit) — это путь между двумя конечными узлами сети. Составной канал образуется отдельными каналами промежуточных звеньев и внутренними соединениями в коммутаторах. Часто эпитет «составной» опускается, и термином «канал» называют как составной канал, так и канал между соседними узлами, то есть в пределах звена.
- **Линия связи** может использоваться как синоним для любого из трех остальных терминов. Линия связи может быть как проводной, так и беспроводной. О беспроводных линиях связи см. главу 23.

Не стоит относиться к путанице в терминологии очень строго. Особенно когда речь идет о различиях в терминологии традиционной телефонии и более новой области — компьютерных сетей. Процесс конвергенции только усугубил проблему терминологии, так как многие механизмы этих сетей стали общими, но сохранили за собой по паре (а иногда и больше) названий, пришедших из каждой области.

Кроме того, существуют объективные причины для неоднозначного понимания терминов. На рис. 6.1 показаны два варианта проводной линии связи. В первом случае линия состоит из сегмента кабеля длиной в несколько десятков метров и представляет собой звено (рис. 6.1, *а*). Во втором случае линия связи представляет собой составной канал, проложенный в сети с коммутацией каналов (рис. 6.1, *б*). Такой сетью может быть **первичная сеть** или телефонная сеть.

Однако для компьютерной сети эта линия представляет собой звено, так как соединяет два соседних узла (например, два маршрутизатора), и вся коммутационная промежуточная аппаратура является прозрачной для этих узлов. Повод для взаимного непонимания на уровне терминов компьютерных специалистов и специалистов первичных сетей здесь очевиден.

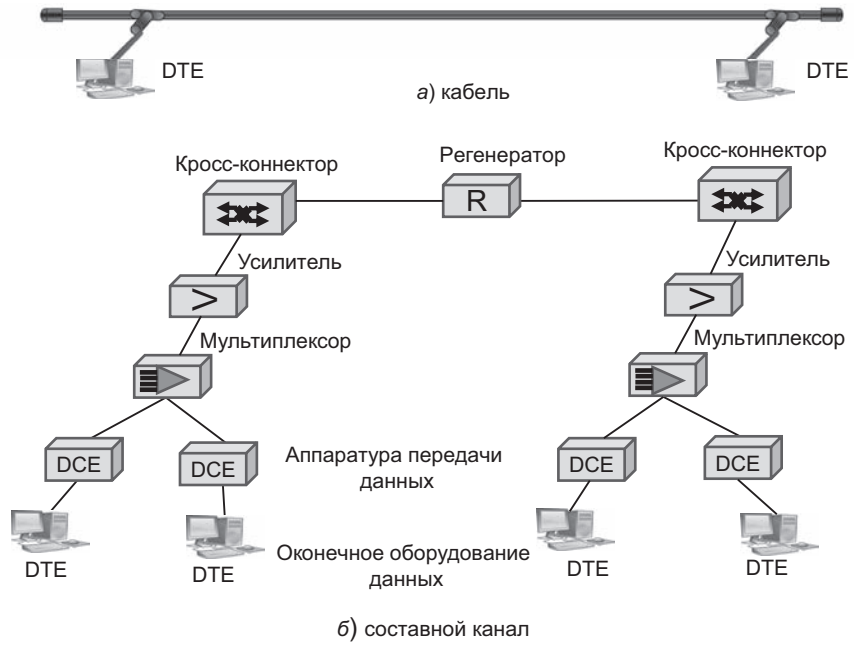


Рис. 6.1. Состав проводной линии связи

Первичные сети предназначены именно для предоставления услуг каналов передачи данных компьютерных и телефонных сетей, про которые в таких случаях говорят, что они работают «поверх» первичных сетей и являются **наложенными сетями**.

Физическая среда передачи данных

Для передачи информации в компьютерных сетях используются **электромагнитные колебания** — взаимосвязанные колебания электрического и магнитного полей. Электромагнитными колебаниями являются радиоволны, микроволны, инфракрасное излучение, видимый свет. Частным случаем электромагнитных колебаний являются **электрические колебания**, когда рассматриваются колебания только электрических величин: силы тока, напряжения, заряда.

Электромагнитные колебания могут распространяться в различных средах, подразделяемых на два класса:

- **направленные среды**, или **проводные** среды. В этом случае электромагнитные волны перемещаются по физически ограниченному пути, например, по медному проводнику (передача электрического напряжения/тока) или волоконно-оптическому волокну. На основе таких проводников строятся проводные (воздушные) или кабельные линии связи (рис. 6.2);
- **ненаправленные среды**, или **беспроводные** среды. Электромагнитные волны распространяются свободно в среде, пропускающей электромагнитное излучение: в земной атмосфере, космическом пространстве, воде, грунте и т. п.

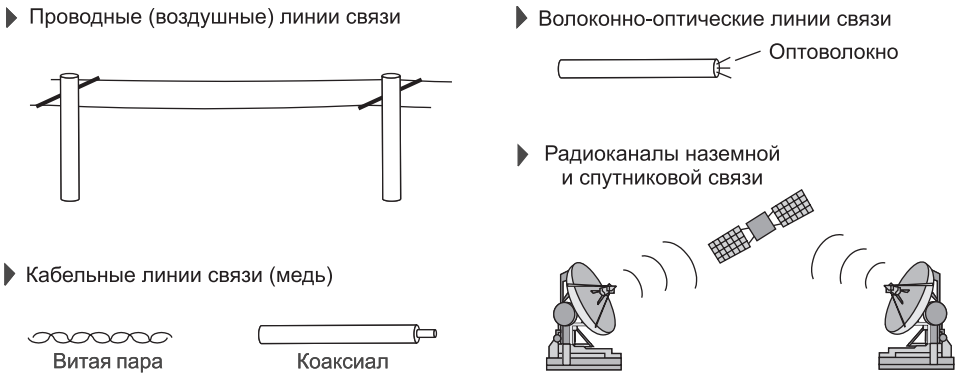


Рис. 6.2. Типы сред передачи данных

Проводные (воздушные) линии связи представляют собой провода без каких-либо изолирующих или экранирующих оплеток, проложенные между столбами и висящие в воздухе. В прошлом такие линии связи были основными для передачи телефонных и телеграфных сигналов. Сегодня проводные линии связи быстро вытесняются кабельными. Но кое-где они все еще сохранились и при отсутствии других возможностей продолжают использоваться, в частности, и для передачи компьютерных данных. Скоростные качества и помехозащищенность этих линий оставляют желать много лучшего.

Кабельные линии имеют достаточно сложную конструкцию. Кабель состоит из набора проводников, заключенных в несколько слоев *изоляции*: электрической, электромагнитной, механической и, возможно, климатической. Кроме того, кабель может быть оснащен *разъемами*, позволяющими быстро выполнять присоединение к нему различного оборудования. В компьютерных (и телекоммуникационных) сетях применяются три основных типа кабеля:

- кабели на основе скрученных пар медных проводов — неэкранированная и экранированная витая пара;
- коаксиальные кабели с медной жилой;
- волоконно-оптические кабели.

Первые два типа кабелей — витую пару и коаксиальный кабель — называют также **медными кабелями**. Волоконно-оптические кабели обладают широкой полосой пропускания и низкой чувствительностью к помехам. На них сегодня строятся как магистрали крупных территориальных и городских сетей, так и высокоскоростные локальные сети.

Радиоканалы наземной и спутниковой связи образуются с помощью передатчика и приемника радиоволн. Существует большое разнообразие типов радиоканалов, отличающихся как используемым частотным диапазоном, так и дальностью канала. Интернет-провайдеры применяют радиоканалы дальнего действия и радиоканалы доступа к мобильным сотовым сетям для предоставления пользователям соединения с интернетом. Беспроводные каналы более короткого действия (Wi-Fi) широко используются для доступа пользователей к устройствам их собственной локальной сети — как подключенной, так и не подключенной к интернету. Более подробно использование радиоканалов рассматривается в части VI.

Аппаратура передачи данных

Как показано на рис. 6.1, линии связи состоят не только из *среды передачи*, но и *аппаратуры*. Даже в том случае, когда линия связи не проходит через первичную сеть, а основана на кабеле, в ее состав входит аппаратура передачи данных.

Аппаратура передачи данных (Data Circuit-terminating Equipment, **DCE**) в компьютерных сетях непосредственно присоединяет компьютеры или коммутаторы к линиям связи и является, таким образом, *пограничным*, или *оконечным, оборудованием линии связи*. Примерами DCE являются модемы (для телефонных линий) и устройства для подключения к цифровым каналам первичных сетей. DCE работает на физическом уровне модели OSI, отвечая за кодирование и передачу информации в физическую среду (в линию) и прием из нее сигналов нужной формы, мощности и частоты.

Аппаратура пользователя линии связи, вырабатывающая *данные* для передачи по линии связи и подключаемая непосредственно к аппаратуре передачи данных, носит обобщенное название **оконечное оборудование данных** (Data Terminal Equipment, **DTE**). Примером DTE могут служить компьютеры, коммутаторы и маршрутизаторы. Эту аппаратуру *не включают* в состав линии связи. Устройства DCE и DTE обычно располагаются на коротких расстояниях друг от друга.

Для подключения DCE-устройств к DTE-устройствам (то есть к компьютерам или коммутаторам/маршрутизаторам) разработаны *стандартные интерфейсы*. Интерфейсы DTE-DCE описываются стандартами серии V CCITT, а также стандартами EIA серии RS. Наиболее популярными стандартами являются RS-232, RS-530, V.35 и HSSI.

ПРИМЕЧАНИЕ

Разделение оборудования на DCE и DTE в локальных сетях является достаточно условным. Например, адаптер локальной сети можно считать как принадлежностью компьютера, то есть оборудованием DTE, так и составной частью канала связи, то есть аппаратурой DCE. Точнее, одна часть сетевого адаптера выполняет функции DTE, а его другая, оконечная часть, непосредственно принимающая и передающая сигналы, относится к DCE.

Помимо оконечной аппаратуры DCE в линию связи может входить **промежуточная аппаратура**. Она решает две основные задачи:

- улучшение качества сигнала;
- создание постоянного составного канала связи между двумя абонентами сети.

В локальных сетях промежуточная аппаратура может совсем не использоваться, если протяженность физической среды — кабелей или радиоэфира — позволяет одному сетевому адаптеру принимать сигналы непосредственно от другого сетевого адаптера без дополнительного усиления. В противном случае применяется промежуточная аппаратура, роль которой здесь играют устройства типа **повторителей** и **концентраторов**.

В *глобальных сетях* необходимо обеспечить качественную передачу сигналов на расстояния в сотни и тысячи километров. Поэтому без **усилителей** (повышающих мощность сигналов) и **регенераторов** (наряду с повышением мощности восстанавливающих форму импульсных сигналов, исказившихся при передаче на большое расстояние), установленных через определенные расстояния, построить территориальную линию связи невозможно.

В первичных сетях, помимо упомянутого оборудования, обеспечивающего качественную передачу сигналов, необходима промежуточная коммутационная аппаратура — **мультиплексоры**, **демультиплексоры** и **кросс-коннекторы** (**коммутаторы**). Эта аппаратура создает между двумя абонентами сети постоянный составной канал из отрезков физической среды — кабелей с усилителями.

В зависимости от типа промежуточной аппаратуры все линии связи делятся на аналоговые и цифровые.

В **аналоговых линиях** промежуточная аппаратура предназначена для усиления аналоговых сигналов, то есть сигналов, которые имеют непрерывный диапазон значений. Такие линии связи традиционно применялись в телефонных сетях с целью связи телефонных коммутаторов между собой. Для создания высокоскоростных каналов, которые мультиплексируют несколько низкоскоростных аналоговых абонентских каналов, при аналоговом подходе обычно используется техника **частотного мультиплексирования** (Frequency Division Multiplexing, **FDM**)[1].

В **цифровых линиях** связи передаваемые сигналы имеют конечное число состояний, то есть являются дискретными. Как правило, элементарный сигнал, то есть сигнал, передаваемый за один такт работы передающей аппаратуры, имеет 2, 3 или 4 состояния, которые в линиях связи воспроизводятся импульсами или потенциалами прямоугольной формы. С помощью таких сигналов передаются как компьютерные данные, так и оцифрованные речь и изображение (именно благодаря одинаковому способу представления информации современными компьютерными, телефонными и телевизионными сетями стало возможным появление общих для всех первичных сетей). В цифровых линиях связи используется специальная промежуточная аппаратура — **регенераторы**, которые улучшают форму импульсов и восстанавливают период их следования. Промежуточная цифровая аппаратура мультиплексирования и коммутации первичных сетей работает по принципу **временно́го мультиплексирования каналов** (Time Division Multiplexing, **TDM**).

Характеристики линий связи

Спектральное представление сигнала

Качество передачи данных по линиям связи определяется как характеристиками передаваемых сигналов (мощность, способ кодирования и др.), так и характеристиками линий связи, к которым относятся: затухание; ограниченность полосы пропускания; помехозащищенность и др. При изучении характеристик линий связи важную роль играет спектральное представление сигналов.

Как известно, процессы могут описываться аналоговыми и дискретными, периодическими и непериодическими функциями (рис. 6.3).

Частным, но фундаментальным случаем аналоговой периодической функции является синусоида.

---

[1] Частотное и временное мультиплексирование рассматриваются в главах 7, 8 и 9.

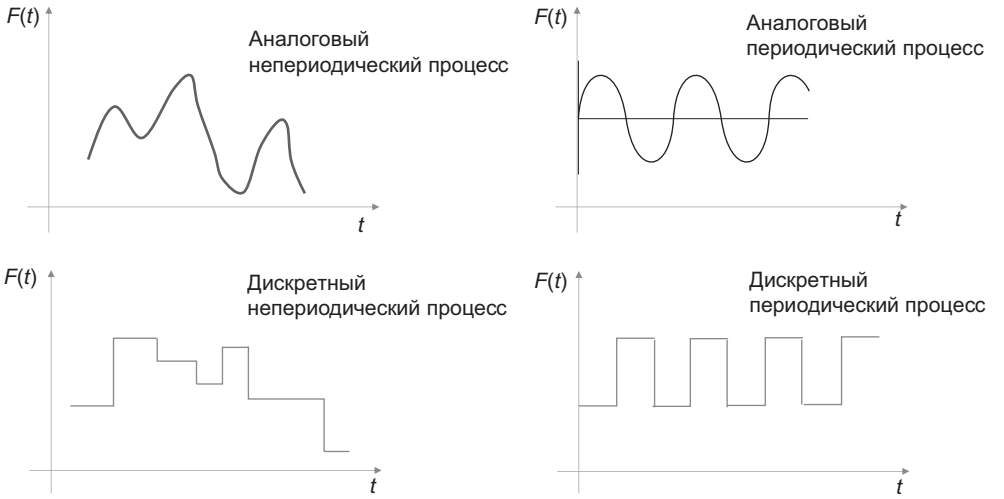


Рис. 6.3. Аналоговые и дискретные, периодические и непериодические функции

Синусоида — волнообразная плоская кривая, которая является графиком периодической тригонометрической функции $y = \sin x$, где y равен отношению противолежащего катета к гипотенузе угла x. Период функции равен 2π, поскольку значение функции $y = \sin x$ при любом x совпадает с ее значением при $x + 2\pi$. Фундаментальное значение этой функции состоит в том, что многие природные процессы описываются этой функцией, например изменение высоты маятника в зависимости от времени, высота волны в жидкости, уровень напряжения в электрической сети.

В общем случае синусоида как *функция от времени* $y(t) = A \sin(2\pi f t + \varphi)$ имеет следующие параметры:

Амплитуда (A) — максимальное значение функции; например, для модулированного сигнала, передаваемого по линии связи, амплитуда равна максимальному уровню напряжения этого сигнала.

Период (T) — время, в течение которого функция выполняет один цикл, $y(t) = y(t + T)$.

Частота (f) — величина, обратная периоду $f = 1/T$. Она также называется *циклической* частотой. Частота определяет, сколько полных колебаний синусоиды происходит за единицу времени, она измеряется в 1/с или в герцах. Коэффициент $2\pi f$ при аргументе t носит специальное название **круговой** или **радианной частоты**, обозначается ω, его можно также выразить через T: $\omega = 2\pi/T$.

Фаза — относительное значение аргумента t в пределах одного периода. Фаза колебания показывает, какая часть периода t/T прошла с момента прохождения синусоидой начальной точки, за которую часто принимают последнее прохождение синусоиды через ноль, при движении из отрицательной в положительную область. Фазу также удобно использовать для описания относительного расположения двух синусоид. Разность фаз двух синусоид называется *сдвигом фаз*. **Начальное значение фазы** φ показывает сдвиг синусоиды от-

носительно начала точки отсчета времени, влево — при положительном значении фазы, вправо — при отрицательном.

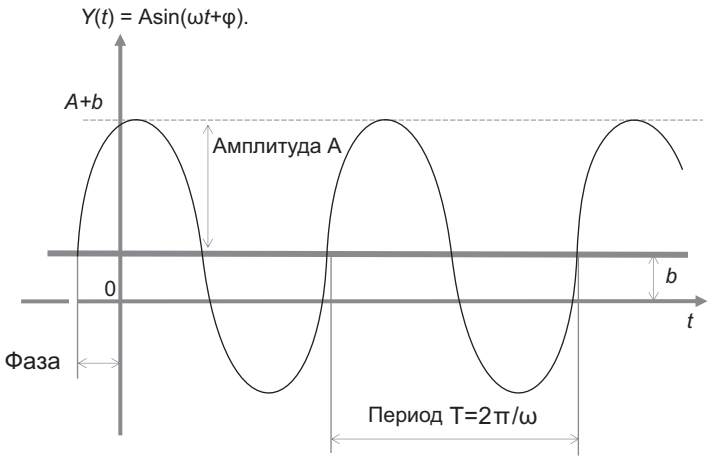


Рис. 6.4. Синусоида

Выше мы рассматривали синусоиду как функцию времени $y(t)$ в некоторой фиксированной точке пространства. Однако можно использовать другое представление синусоидальной функции $y(x)$, когда ее значения изменяются в зависимости от расстояния x от некоторой точки. В реальной жизни мы часто сталкиваемся именно с такой природой колебательных процессов, когда колебания способны перемещаться, удаляясь от места возникновения. Синусоида как функция времени составляет единое целое с синусоидой — функцией расстояния, то есть является функцией двух переменных $y(t, x)$, что отражает взаимосвязь временной и пространственной периодичности.

У синусоиды $y(x)$ имеется параметр — длина волны, который является аналогом периода синусоиды $y(t)$. **Длина волны** (λ) определяется как расстояние, на которое перемещается волна за время, равное периоду T, отсюда скорость распространения волны υ в данной среде равна $\upsilon = \lambda/T$. В вакууме все электромагнитные волны распространяются со скоростью, равной 300 000 км/с, независимо от их частоты. Эта скорость, обозначаемая как c (си), называется **скоростью света**. Таким образом, для вакуума справедливо фундаментальное соотношение $c = \lambda/T$ или $c = \lambda f$.

Синусоидальные функции обладают многими свойствами, делающими их эффективным инструментом изучения сигналов и линий связи. Одной из таких особенностей, например, является связанный с ними развитый математический аппарат. Поэтому такую важность обретает факт, что процесс, описываемый *произвольной, не обязательно синусоидальной* функцией, может быть представлен в виде некоторого набора синусоидальных функций.

Из теории гармонического анализа Фурье известно, что любой периодический процесс можно представить в виде суммы бесконечного набора синусоидальных колебаний различных частот и различных амплитуд. Этот набор называют **спектральным разложением, разложением Фурье** или **спектром**, а синусоидальное колебание определенной частоты — **гармоникой**.

Математически разложение Фурье выглядит так:

$$f(t) = a_0 + \sum_{n=1}^{\infty}[a_n \cos(n\omega_0 t) + b_n \sin(n\omega_0 t)],$$

где $\omega_0 = 1/T$,

с коэффициентами

$$a_0 = 1/T \int_T f(t)\mathrm{d}t,$$

$$a_n = 2/T \int_T f(t)\cos(n\omega_0 t)\mathrm{d}t,\ b_n = 2/T \int_T f(t)\sin(n\omega_0 t)\mathrm{d}t,$$

где интегрирование проводится по любому интервалу времени длительностью T.

Нахождение амплитуд гармоник a_n и b_n по приведенным выше формулам называется **прямым преобразованием Фурье**, а операция восстановления непрерывного периодического сигнала $f(t)$ по известным величинам гармоник — **обратным преобразованием Фурье**.

Сегодня аналоговые сигналы эффективно обрабатываются цифровыми процессорами. Для выполнения прямого преобразования Фурье с помощью цифрового процессора аналоговый сигнал сначала оцифровывается, а затем коэффициенты разложения Фурье вычисляются заменой интегралов на конечные суммы. Цифровые процессоры обычно реализуют алгоритм **быстрого преобразования Фурье (Fast Fourier Transformation, FTT)**, который дает значительное сокращение вычислений коэффициентов разложения, например, примерно в 100 раз при представлении исходной функции 1024 точками. Для обратного преобразования Фурье также имеется алгоритм его быстрого вычисления — **IFTT (Inverse Fast Fourier Transformation)**.

На рис. 6.5 показаны первые четыре гармоники спектрального разложения периодической аналоговой функции, найденные с помощью прямого преобразования Фурье. Для восстановления исходного периодического сигнала по его спектру, то есть выполнения обратного преобразования Фурье, необходимо просуммировать *все* гармоники его разложения.

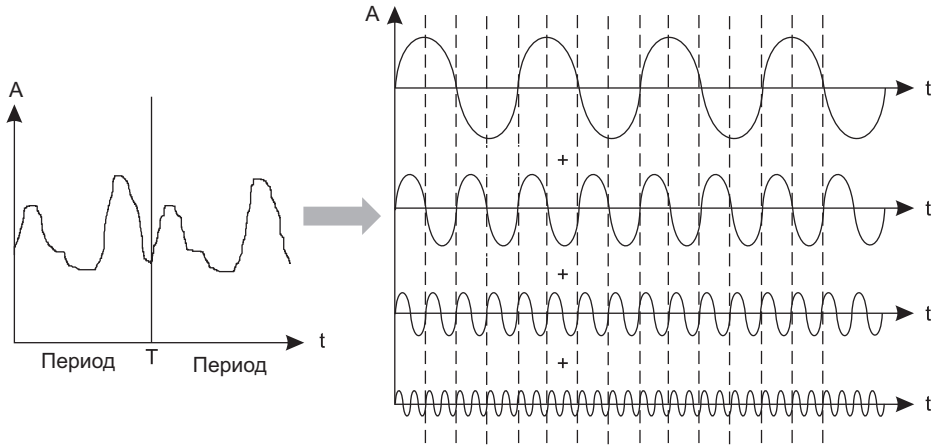


Рис. 6.5. Представление периодического аналогового сигнала суммой синусоид

Однако обычно во внимание принимается только несколько первых, так называемых **значимых гармоник**, потому что амплитуды последующих гармоник быстро убывают и вносят незначительный вклад в форму исходного сигнала. Самая первая частота спектра называется **основной гармоникой**.

Разность между максимальной и минимальной частотами значимого набора синусоид, которым представлен исходный сигнал, называется **шириной спектра сигнала**.

Импульсные сигналы, наряду с другими способами кодирования дискретных компьютерных данных, часто используются в каналах связи, поэтому их спектральное разложение представляет для нас особый интерес. На рис. 6.6 показан периодический дискретный сигнал — последовательность прямоугольных импульсов. Этот сигнал может быть представлен суммой бесконечного числа синусоид, причем каждая из последующих имеет меньшую амплитуду и период. На этом рисунке мы видим две первые гармоники.

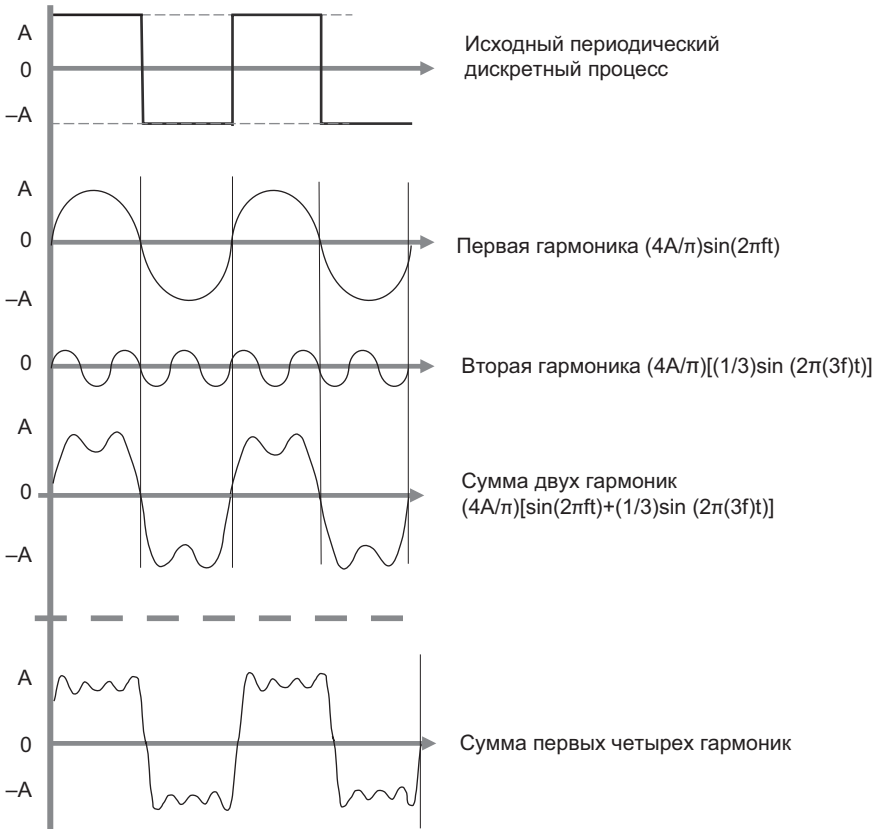


Рис. 6.6. Разложение и восстановление прямоугольных импульсов на основе спектрального представления

Если их просуммировать, то в результате можно увидеть последовательность искаженных, но вполне распознаваемых прямоугольных импульсов. Еще ближе к исходному сигналу результат суммирования первых четырех гармоник.

Отдельные гармоники можно представлять либо в виде *синусоидальных функций* от времени/расстояния, как это показано на рис. 6.6, либо в виде *точек* в координатах «частота — амплитуда», как на рис. 6.7 (здесь также изображен спектр последовательности прямоугольных импульсов).

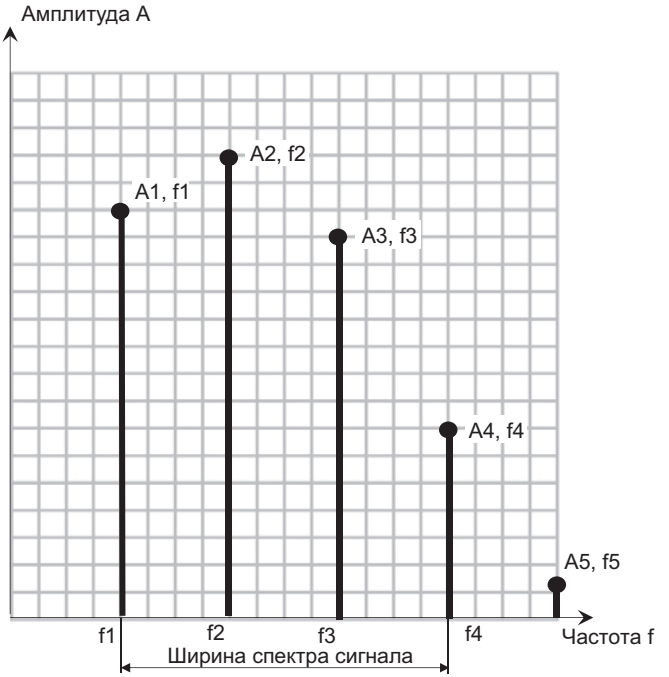


Рис. 6.7. Точечный спектр периодической последовательности прямоугольных импульсов

Непериодические процессы также можно представить в виде бесконечной непрерывной суммы синусоид *всех возможных частот*, то есть в виде *интеграла*. Для сигналов произвольной формы, встречающихся на практике, спектр можно найти с помощью специальных приборов — **спектральных анализаторов**, которые измеряют спектр реального сигнала и отображают амплитуды составляющих гармоник на экране (рис. 6.8), распечатывают их на принтере или передают для обработки и хранения в компьютер в оцифрованном виде.

Быстрые прямое и обратное преобразования Фурье применяются во многих областях телекоммуникационных технологий: анализе спектра модулированного сигнала и генерации таких сигналов, детектировании сигналов на фоне помех на основе корреляционного анализа спектра сигнала, конструировании цифровых фильтров и др.

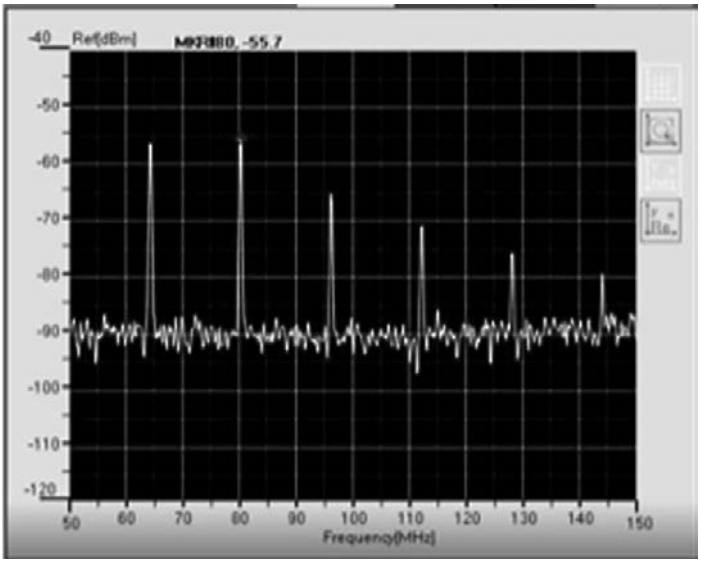


Рис. 6.8. Спектр сигнала, найденного спектральным анализатором

Затухание и опорная мощность

Разложение произвольного информационного сигнала на гармоники дает нам возможность теоретически *предсказать* его изменение (искажение) при прохождении среды передачи данных, что очень важно при проектировании линий связи, предполагающем выбор передающей среды (например, медный проводник, оптическое волокно или радиоволны), способ кодирования информации, мощность передатчика, чувствительность приемника.

Это предсказание основывается на следующем замечательном свойстве синусоидального сигнала — *при прохождении через однородную распределенную в пространстве среду он сохраняет свою синусоидальную форму и частоту, изменяется только его амплитуда и фаза.*

Следовательно, если мы будем знать, как некоторая среда передает синусоиды различной частоты, то сможем предсказать, как эта среда передаст любой периодический сигнал, в связи с чем не придется тестировать некоторую среду каждый раз, когда мы решим применить сигнал некоторой новой формы. Нам достаточно будет найти разложение Фурье для нашего нового сигнала и, зная, как среда передает гармоники, просуммировать или проинтегрировать их и получить результирующий сигнал на выходе среды. То есть зависимость амплитуды передаваемых гармоник от частоты для определенной среды должна быть найдена только один раз, а затем ее можно применять для анализа передачи различных типов сигналов этой средой.

Для нахождения этой зависимости необходимо протестировать данную среду набором *эталонных синусоидальных сигналов* заданной амплитуды, но различной частоты, изменяя ее с некоторым шагом. Генерация синусоидального сигнала не представляет собой трудности, поскольку он присущ многим физическим процессам, например, переменному электрическому току или звуковым или световым волнам. Найденная зависимость изменения амплитуды сигнала после прохождения среды для ряда последовательных частот называется **амплитудно-частотной характеристикой среды**.

Построив амплитудно-частотную характеристику среды, мы можем применить ее к спектральному разложению сигнала, то есть найти, какую амплитуду будет иметь каждая из составляющих нашего периодического сигнала после прохождения среды. Затем следует сложить все составляющие (гармоники) и получить результирующий выходной сигнал. Заметим, что мы опустили в этом рассуждении изменение фазы сигнала при его прохождении через среду. Подобное упрощение допустимо, если фазы различных гармоник изменяются одинаково, что часто бывает на практике. В противном случае изменение фаз нужно принимать во внимание с учетом данных о фазо-частотной характеристике среды для анализа искажений сигнала.

На практике в качестве характеристики линии связи чаще используется не зависимость амплитуды от частоты, а зависимость *мощности* сигнала от частоты. Обе зависимости дают одну и ту же качественную картину, так как мощность синусоидального сигнала пропорциональна квадрату его амплитуды, но существующие спектральные анализаторы более приспособлены измерять не амплитуду сигнала, а его мощность.

Уменьшение мощности сигнала на пути от источника к приемнику кажется интуитивно понятным. Если обратиться к интересующим нас типам сигналов — прямоугольным импульсам, то эффект затухания (упрощенно, без учета других искажений) выражается в уменьшении амплитуды импульсов (рис. 6.9).

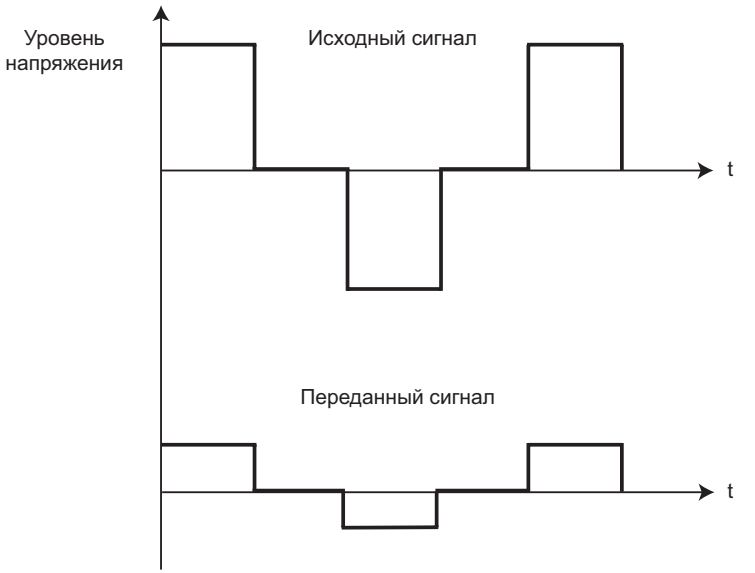


Рис. 6.9. Влияние затухания на форму прямоугольного импульса

Затухание показывает, насколько уменьшается мощность эталонного синусоидального сигнала на выходе линии связи по отношению к мощности сигнала на входе этой линии. Затухание (A) обычно измеряется в децибелах (дБ) и вычисляется по следующей формуле:

$$A = 10 \lg P_{out}/P_{in}$$

Здесь P_{out} мощность сигнала на выходе линии, P_{in} - мощность сигнала на входе линии, измеряемые в ваттах.

Знание затухания линии очень важно, так как оно позволяет оценить необходимую мощность передатчика и чувствительность приемника (то есть минимальную мощность сигнала, которую приемник устойчиво распознает).

Затухание зависит от длины линии связи, поэтому в качестве характеристики линии связи используется так называемое **погонное затухание**, то есть затухание на линии связи определенной длины. Для кабелей локальных сетей в качестве такой длины обычно используют 100 м (это значение является максимальной длиной кабеля для многих технологий LAN). Для территориальных линий связи погонное затухание измеряют для расстояния в 1 км.

Обычно затуханием характеризуют пассивные участки линии связи, состоящие из кабелей и кроссовых секций, без усилителей и регенераторов. Так как мощность выходного сигнала кабеля без промежуточных усилителей меньше, чем мощность входного, затухание кабеля всегда является *отрицательной величиной*.

Чаще всего при описании параметров линии связи приводятся значения затухания всего *для нескольких значений частот*. Это объясняется, с одной стороны, стремлением упростить измерения при проверке качества линии. С другой стороны, на практике часто заранее известна основная частота передаваемого сигнала, то есть та частота, гармоника которой имеет наибольшую амплитуду и мощность. Поэтому достаточно знать затухание на этой частоте, чтобы приблизительно оценить искажения передаваемых по линии сигналов.

ВНИМАНИЕ

Затухание всегда имеет отрицательное значение, однако знак «минус» часто опускают, из-за чего порой возникает путаница. Совершенно корректно утверждение, что качество линии связи тем выше, чем больше (с учетом знака) затухание. Если же игнорировать знак, то есть иметь в виду абсолютное значение затухания, то у более качественной линии затухание меньше, и это наиболее распространенная форма интерпретации этого термина.

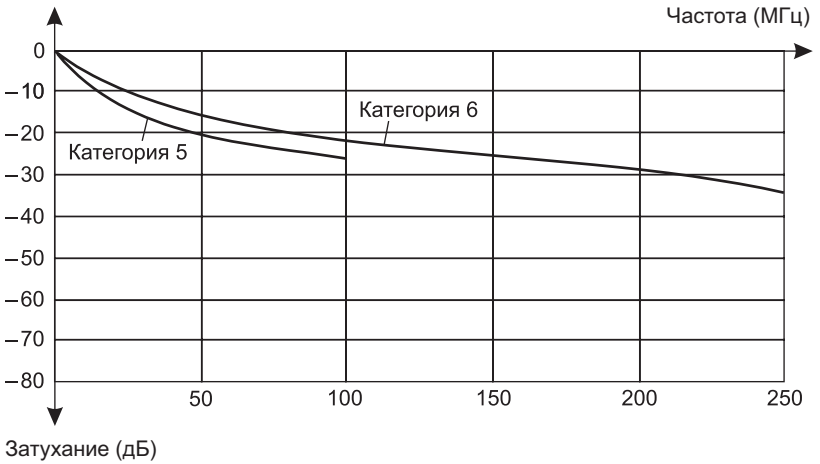


Рис. 6.10. Затухание неэкранированного кабеля на витой паре

На рис. 6.10 показаны типовые зависимости затухания от частоты для кабелей длиной 100 м на неэкранированной витой паре категорий 5 и 6. Чем выше категория кабеля, тем он

качественнее (см. графики затухания). Так, кабель категории 5 имеет на частоте 100 МГц затухание –23,6 дБ, а у более качественного кабеля категории 6 на этой же частоте затухание равно –20,6 дБ. Из графиков также видно, что с повышением частоты сигнала затухание увеличивается. Отсюда проистекают проблемы передачи дискретных данных с высокой скоростью: как мы увидим далее, при возрастании скорости передачи сигнала его частота увеличивается, что приводит и к росту затухания.

Оптический кабель имеет существенно меньшие (по абсолютной величине) величины затухания — обычно в диапазоне от –0,2 до –3 дБ при длине кабеля в 1000 м, а значит, является более качественным, чем кабель на витой паре. Практически для всех оптических волокон типична сложная зависимость затухания от длины волны, которая имеет три так называемых **окна прозрачности**. На рис. 6.11 показана характерная зависимость затухания для оптического волокна (в оптике принято игнорировать знак затухания, поэтому ось величины затухания идет вверх). Из рисунка видно, что область эффективного использования современных волокон ограничена волнами длин 850 нм, 1310 нм и 1550 нм с шириной каждого окна примерно 100 нм (соответственно, частотами 35 ТГц, 23 ТГц и 19,4 ТГц). Окно 1550 нм обеспечивает наименьшие потери и, следовательно, максимальную дальность при фиксированной мощности передатчика и фиксированной чувствительности приемника.

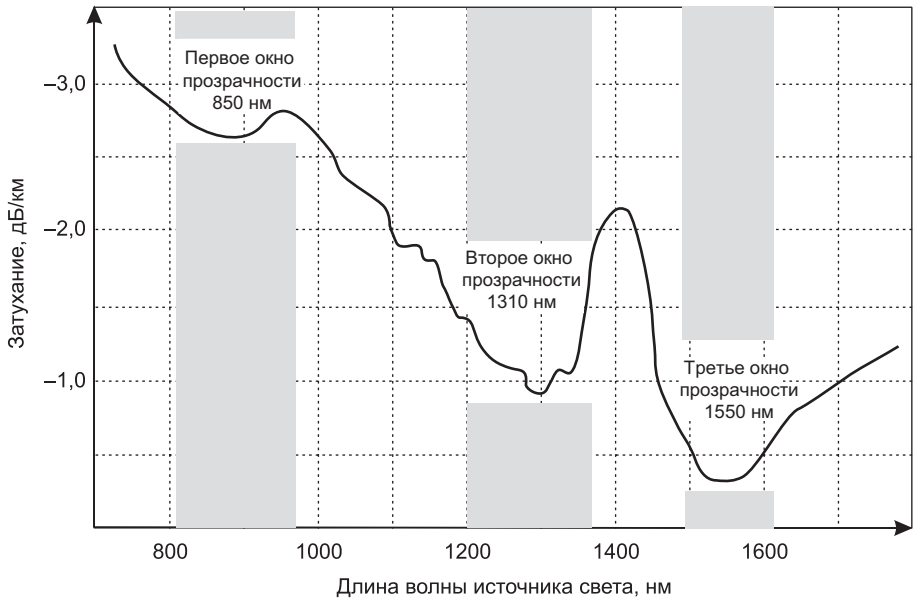


Рис. 6.11. Окна прозрачности оптического волокна

В качестве характеристики мощности сигнала используются абсолютный и относительный уровни мощности. **Абсолютная мощность** измеряется в ваттах, **относительная мощность** — в децибелах и рассчитывается по той же формуле, что и затухание. Такая величина, как относительная мощность, используется при сравнении двух сигналов, например сигнала помехи и информационного сигнала. Затухание также является примером относительной мощности, в этом случае мы сравниваем мощность сигнала на выходе и входе линии связи.

Частным случаем относительной мощности является **опорная мощность**. При расчете опорной мощности уровень, на который делится измеряемая мощность, принимается равным 1 мВт, что и отражается в названии этой единицы мощности, *децибел на милливатт* (дБм).

Опорная мощность p вычисляется по формуле:

$$p = 10 \lg P/1 \text{ мВт [дБм]}.$$

Здесь P — абсолютная мощность сигнала в милливаттах.

Несмотря на использование отношения в определении опорной мощности, эта единица измерения является *абсолютной*, а не относительной, так как однозначно преобразует абсолютную мощность сигнала в ваттах в некоторое значение, которое никак не зависит от значения мощности другого сигнала, как это имеет место при определении децибела. Так, нетрудно вычислить соответствие некоторых значений мощности сигнала, выраженные в ваттах и дБм:

$$1 \text{ мВ} = 0 \text{ дБм}, 10 \text{ мВ} = 10 \text{ дБм}, 1 \text{ В} = 30 \text{ дБм}, 100 \text{ кВ} = 80 \text{ дБм}.$$

Опорные значения мощности удобно использовать при *расчетах энергетического бюджета линий связи.*

Энергетическим бюджетом линии (link budget) называется расчет баланса всех усилений и ослаблений сигнала на пути от передатчика к приемнику. Бюджет линии позволяет оценить уровень мощности сигнала, пришедшего на вход приемника, и понять, достаточен ли он для надежного распознавания передаваемых по линии сигналов.

Пример

Пусть требуется определить минимальную опорную мощность x (дБм) передатчика, достаточную для того, чтобы на выходе линии опорная мощность сигнала была не ниже некоторого порогового значения y (дБм). Затухание линии известно и равно A. Пусть X и Y — это абсолютные значения мощности сигнала, заданные в милливаттах на входе и выходе линии соответственно.

По определению $A = 10 \lg X/Y$. Используя свойства логарифмов, приходим к следующему:

$$A = 10 \lg X/Y = 10\lg(X/1)/(Y/1) = 10 \lg X/1 \text{ мВт} - 10 \lg Y/1 \text{ мВт}.$$

Заметим, что два последних члена уравнения по определению являются опорными значениями мощности сигналов на выходе и входе. Отсюда приходим к простому соотношению $A = x - y$, где x — опорная мощность входного сигнала, а y — опорная мощность выходного сигнала.

Следовательно, минимальная требуемая мощность передатчика может быть определена как сумма затухания и опорной мощности сигнала на выходе: $x = A + y$.

Величина опорной мощности y выходного сигнала, являющаяся минимальной опорной мощностью сигнала на входе приемника, при котором он еще способен корректно распознавать дискретную информацию, содержащуюся в сигнале, называется **порогом чувствительности приемника**. Для нормальной работы линии связи необходимо, чтобы минимальная опорная мощность сигнала передатчика, даже ослабленная затуханием линии связи, превосходила порог чувствительности приемника: $x - A > y$. Проверка этого условия и является сутью расчета энергетического бюджета линии.

Полоса пропускания

На форму сигнала, передаваемого по линии связи, влияет ограниченность ее полосы пропускания.

Полоса пропускания (bandwidth) — это непрерывный диапазон частот, для которого затухание не превышает некоторый заранее заданный предел. То есть полоса пропускания определяет диапазон частот синусоидального сигнала, при которых этот сигнал передается по линии связи без значительных искажений. Полоса пропускания измеряется в герцах (Гц).

ПРИМЕЧАНИЕ

Термин «полоса пропускания» иногда используется как синоним термина «пропускная способность» и измеряется в битах в секунду. Конечно, лучше было бы применять разные термины для описания различных характеристик, но существуют традиции, которые изменить трудно. В таких случаях приходится различать значения термина по контексту.

Степень затухания мощности синусоидального сигнала зависит от частоты синусоиды. Эта зависимость также характеризует линию связи (рис. 6.12).

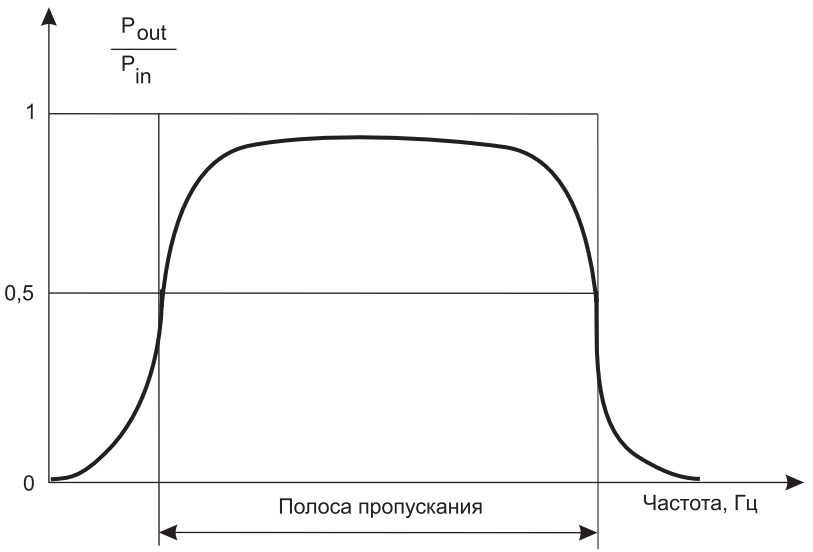


Рис. 6.12. Зависимость затухания от частоты

Часто граничными частотами полосы пропускания считаются частоты, на которых мощность выходного сигнала уменьшается в два раза по отношению к входному, что соответствует затуханию в –3 дБ. Как мы увидим далее, *ширина* полосы пропускания в наибольшей степени влияет на максимально возможную скорость передачи информации по линии связи.

Искажение передающей линией связи синусоиды какой-либо частоты в конечном счете приводит к искажению амплитуды и формы передаваемого сигнала, так как гармоники

различных частот искажаются *неодинаково*. Если это аналоговый сигнал, передающий речь, то изменяется тембр голоса за счет искажения обертонов — боковых частот.

При передаче импульсных сигналов, характерных для компьютерных сетей, искажаются низкочастотные и высокочастотные гармоники. В результате фронты импульсов теряют свою прямоугольную форму (рис. 6.13) и сигналы могут плохо распознаваться на приемном конце линии.

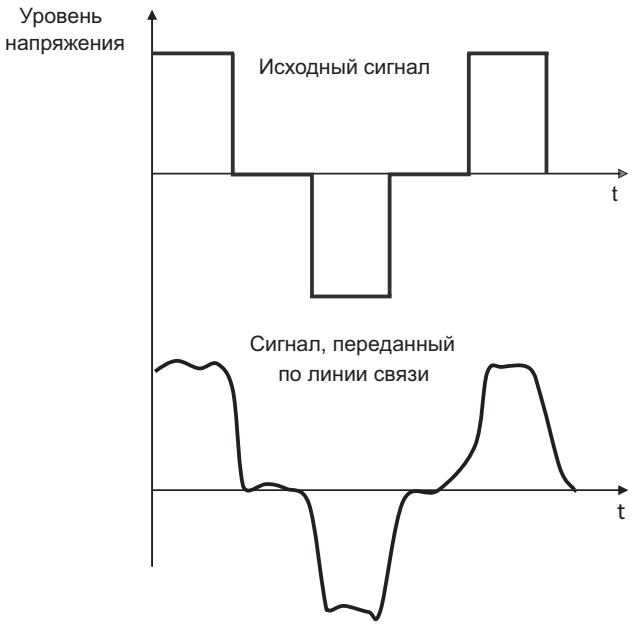


Рис. 6.13. Влияние ограниченности полосы пропускания на форму прямоугольного импульса

Помехи

Передаваемые сигналы искажаются из-за несовершенства линий связи, а также из-за внешних и внутренних помех.

Для электрических сигналов *идеальная передающая среда*, не вносящая никаких помех в передаваемый сигнал, должна, по меньшей мере, иметь нулевые значения сопротивления, емкости и индуктивности. Однако на практике медные провода, например, всегда представляют собой некоторую распределенную по длине комбинацию активного сопротивления, емкостной и индуктивной нагрузок (рис. 6.14). В результате синусоиды различных частот передаются этими линиями по-разному.

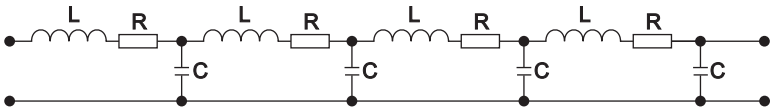


Рис. 6.14. Представление медной линии как распределенной индуктивно-емкостной нагрузки

Важным параметром медной линии связи является ее **волновое сопротивление**, представляющее собой полное (комплексное) сопротивление, которое встречает электромагнитная волна определенной частоты при распространении вдоль однородной цепи. Волновое сопротивление измеряется в омах и зависит от таких параметров линии связи, как активное сопротивление, погонная индуктивность и погонная емкость, а также от частоты самого сигнала. Выходное сопротивление передатчика должно быть согласовано с волновым сопротивлением линии, иначе затухание сигнала будет неприемлемым.

Помимо искажений сигналов, возникающих из-за неидеальных физических параметров линии связи, свой вклад в искажение формы сигналов на выходе линии вносят помехи.

Электромагнитная помеха — это нежелательное воздействие на передаваемый сигнал, ухудшающее возможность его распознавания при приеме.

В результате помех сигналы на выходе линии связи могут иметь искаженную форму (рис. 6.15). Источник помех может находиться вне или внутри линии связи.

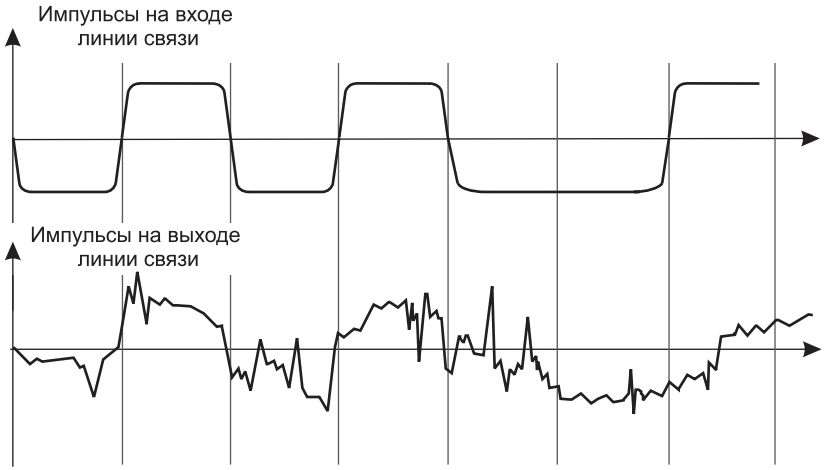


Рис. 6.15. Искажение импульсов в линии связи

Внешние помехи создаются различными электрическими двигателями, электронными устройствами, атмосферными явлениями и т. д. **Внутренние помехи** — это результат взаимного влияния электрических процессов в проводниках линии связи, так называемые перекрестные **наводки** одной пары проводников на другую.

Помеха может быть как *регулярной*, так и *случайной* величиной. В первом случае она может быть достаточно легко устранена. Например, наводка от силовой линии переменного тока может быть компенсирована соответствующим противофазным сигналом, а помеха от определенной радиостанции известной частоты может быть заблокирована фильтром, не пропускающим эту частоту. Сложнее обстоит дело со случайной помехой, которую нельзя заранее предсказать и скорректировать. Несмотря на защитные меры, предпринимаемые разработчиками кабелей, наличие усилительной и коммутирующей аппаратуры, полностью компенсировать влияние случайных внешних помех, как правило, не удается.

Разные виды помех могут по-разному воздействовать на исходный сигнал. В том случае, когда результирующий сигнал может быть представлен в виде суммы исходного сигнала и помехи, помеху называют **аддитивной помехой** или **шумом** (рис. 6.16).

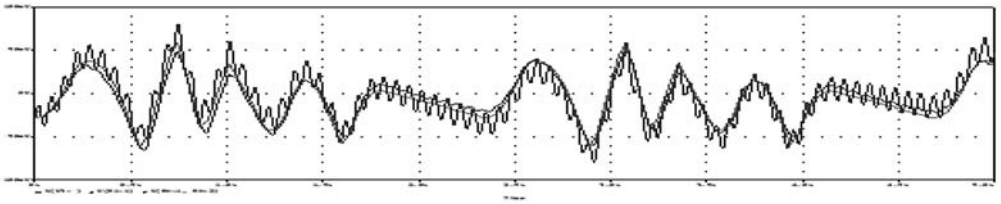


Рис. 6.16. Влияние аддитивной помехи на исходный сигнал

Важнейшим параметром, характеризующим работу линии связи в условиях аддитивных помех, является отношение **«сигнал/шум»** (signal-to-noise ratio, **SNR**), равное отношению средней мощности исходного сигнала к средней мощности шума $P_c/P_{ш}$. Чем меньше это значение, тем более сложным должно быть устройство приемника, чтобы он мог компенсировать помехи. Как мы увидим далее, это соотношение занимает центральное место в формуле Шеннона, описывающей связь между пропускной способностью и полосой пропускания.

Помехоустойчивость линии, как и следует из названия, определяет способность линии противостоять влиянию помех, создаваемых во внешней среде или на внутренних проводниках самого кабеля. Помехоустойчивость линии зависит от типа используемой физической среды, а также от средств экранирования и подавления помех самой линии. Наименее помехоустойчивыми являются радиолинии, хорошей устойчивостью обладают кабельные линии и отличной — волоконно-оптические линии, малочувствительные к внешнему электромагнитному излучению. Обычно для уменьшения помех, создаваемых внешними электромагнитными полями, проводники экранируют и/или скручивают.

Электрическая и магнитная связь — это параметры медного кабеля, характеризующие воздействие на него помех. **Электрическая связь** определяется отношением наведенного тока в цепи, подверженной влиянию, к напряжению, действующему во влияющей цепи. **Магнитная связь** — это отношение электродвижущей силы, наведенной в подверженной влиянию цепи, к току во влияющей цепи. Результатом электрической и магнитной связи являются **наведенные сигналы** (наводки) в цепи, подверженной влиянию. Существует несколько различных параметров, характеризующих устойчивость кабеля к наводкам.

Перекрестные наводки на ближнем конце (Near End Cross Talk, NEXT) определяют устойчивость кабеля в том случае, когда наводка образуется в результате действия сигнала, генерируемого передатчиком, подключенным к одной из соседних пар на том же конце кабеля, на котором работает подключенный к подверженной влиянию паре приемник (рис. 6.17). Показатель NEXT, выраженный в децибелах, равен $10 \lg P_{out}/P_{ind}$, где P_{out} — мощность выходного сигнала, P_{ind} — мощность наведенного сигнала.

Чем меньше значение NEXT, тем лучше кабель. Так, для витой пары категории 5 показатель NEXT должен быть меньше –27 дБ на частоте 100 МГц.

Перекрестные наводки на дальнем конце (Far End Cross Talk, FEXT) позволяют оценить устойчивость кабеля к наводкам для случая, когда передатчик и приемник подключены к разным концам кабеля. Очевидно, что этот показатель должен быть лучше, чем NEXT, так как до дальнего конца кабеля сигнал приходит ослабленным затуханием каждой пары.

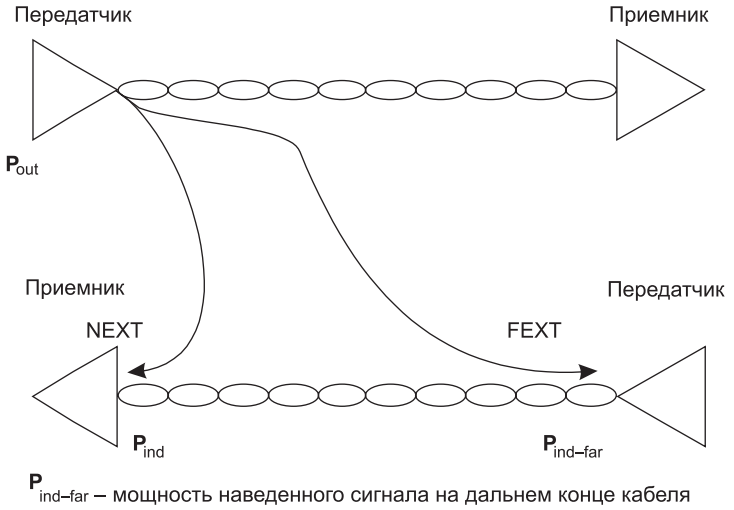


Рис. 6.17. Переходное затухание

Показатели NEXT и FEXT обычно применяются к кабелю, состоящему из нескольких витых пар, так как в этом случае взаимные наводки одной пары на другую могут достигать значительных величин. Для одинарного коаксиального кабеля (то есть состоящего из одной экранированной жилы) этот показатель не имеет смысла, а для двойного коаксиального кабеля он также не применяется вследствие высокой степени защищенности каждой жилы. Оптические волокна тоже не создают сколько-нибудь заметных взаимных помех.

Еще одной характеристикой линии связи является **достоверность передачи данных**. Она характеризует вероятность искажения каждого передаваемого бита данных. Иногда этот же показатель называют **интенсивностью битовых ошибок** (Bit Error Rate, BER). Величина BER для линий связи без дополнительных средств защиты от ошибок (например, самокорректирующихся кодов или протоколов с повторной передачей искаженных кадров) составляет, как правило, 10^{-4}–10^{-6}, в оптоволоконных линиях связи — 10^{-9}. Значение достоверности передачи данных в 10^{-4}, к примеру, говорит о том, что в среднем из 10 000 бит искажается значение одного бита.

Пропускная способность

Пропускная способность, называемая также **емкостью** линии связи (capacity), характеризует максимальную скорость передачи данных, которая может быть достигнута на этой линии.

Особенностью пропускной способности является то, что, с одной стороны, она зависит от характеристик *физической среды* (затухания и полосы пропускания), а с другой — *способа передачи данных* (кодирования). Следовательно, нельзя говорить о пропускной способности линии связи до того, как для нее определен протокол физического уровня.

Например, если для линии связи определен протокол физического уровня, задающий фиксированную **битовую скорость передачи данных** (bit rate of transmitter), то для нее известна и соответствующая пропускная способность — например, 2 Мбит/с, 100 Мбит/с, 1 Гбит/с и т. п.

В тех же случаях, когда только предстоит выбрать, какой из множества существующих протоколов использовать на данной линии, очень важными являются остальные характеристики линии: полоса пропускания, перекрестные наводки, помехоустойчивость и др.

Пропускная способность, как и скорость передачи данных, измеряется в битах в секунду (бит/с), а также в производных единицах: килобит в секунду (Кбит/с) и т. д.

ВНИМАНИЕ

Пропускная способность линий связи и коммуникационного сетевого оборудования традиционно измеряется в битах в секунду, а не в байтах в секунду. Это связано с тем, что данные в сетях передаются последовательно, то есть побитно, а не параллельно, байтами, как это происходит между устройствами внутри компьютера. Такие единицы измерения, как килобит, мегабит или гигабит, в сетевых технологиях строго соответствуют степеням десяти (то есть килобит — это 1000 бит, а мегабит — это 1 000 000 бит), как это принято во всех отраслях науки и техники, а не близким к этим числам степеням двойки, как это принято в программировании, где приставка «кило» равна 2^{10} = 1024, а «мега» — 2^{20} = 1 048 576.

Пропускная способность линии связи зависит не только от таких ее характеристик, как затухание и полоса пропускания, но и от спектра передаваемых сигналов. Если значимые гармоники сигнала (то есть те гармоники, амплитуды которых вносят основной вклад в результирующий сигнал) попадают в полосу пропускания линии, то такой сигнал будет хорошо передаваться данной линией связи, и приемник сможет правильно распознать информацию, отправленную по линии передатчиком (рис. 6.18, *а*). Если же значимые гармоники выходят за границы полосы пропускания линии связи, то сигнал будет значительно искажаться, что усложнит приемнику распознавание информации (рис. 6.18, *б*).

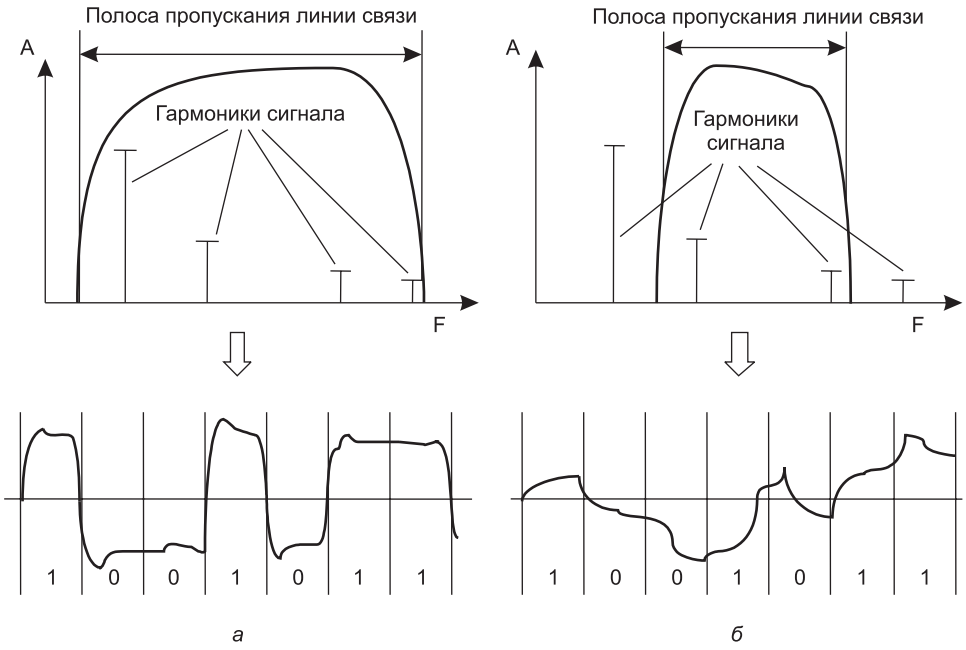


Рис. 6.18. Соответствие между полосой пропускания линии связи и спектром сигнала

Влияние способа кодирования на пропускную способность

Выбор способа представления дискретной информации в виде сигналов, подаваемых на линию связи, называется **физическим** или **линейным кодированием**. От выбранного способа кодирования зависит спектр сигналов и, соответственно, пропускная способность линии.

Таким образом, для одного способа кодирования линия может обладать одной пропускной способностью, а для другого — другой. Например, витая пара категории 5 может передавать данные с пропускной способностью 100 Мбит/с при способе кодирования стандарта физического уровня 100Base-T, и 1 Гбит/с при способе кодирования стандарта 1000Base-T.

> В соответствии с основным постулатом теории информации *любое различимое непредсказуемое* изменение принимаемого сигнала несет в себе информацию. Отсюда следует, что синусоида, у которой амплитуда, фаза и частота остаются неизменными, информации не несет, так как изменение сигнала хотя и происходит, но является абсолютно предсказуемым. Аналогично, не несут в себе информации импульсы на тактовой шине компьютера, так как их изменения тоже постоянны во времени. А вот импульсы на шине данных предсказать заранее нельзя, что и делает их информационными — они переносят информацию между отдельными блоками или устройствами компьютера.

В большинстве способов кодирования используется изменение одного или нескольких параметров периодического электрического сигнала — частоты, амплитуды и фазы синусоиды или же уровня напряжения/тока последовательности импульсов. Эти параметры называют **информационными параметрами сигнала**. Периодический сигнал, параметры которого подвергаются изменениям, называют **несущим сигналом**. Процесс изменения информационных параметров несущего сигнала в соответствии с передаваемой информацией называется **кодированием** или **модуляцией**. Измененный в результате кодирования несущий сигнал называют **информационным сигналом**.

Если информационный сигнал изменяется так, что можно различить только два его состояния, то любое его изменение будет соответствовать наименьшей единице информации — биту. Если же сигнал может иметь более двух различимых состояний, то любое его изменение будет нести *несколько битов информации*. Например, для сигнала с четырьмя состояниями одно изменение несет два бита: 00, 01, 10, 11.

Передача дискретной информации в телекоммуникационных сетях осуществляется тактированно, то есть изменение информационного параметра сигнала происходит через фиксированный интервал времени, называемый **тактом**. Величина, обратная значению такта, является **тактовой частотой** линии. Передатчик может отдельно от информационных сигналов передавать тактовые сигналы, например электрические импульсы, по отдельной линии связи, чтобы приемнику было легче распознавать начало каждого такта. Возможна также схема работы приемника и передатчика без передачи отдельного тактового сигнала — в этом случае приемник должен обладать очень точным таймером, а также возможностью иногда подстраивать свой таймер под таймер передатчика за счет анализа информационного сигнала, например, если информационный сигнал является прямоугольным импульсом, то подстройка может выполняться при приходе переднего фронта импульса.

Приемник информации считает, что в начале каждого такта на его вход поступает новая информация. При этом независимо от того, повторяет ли сигнал состояние предыдущего

такта или же он имеет состояние, отличное от предыдущего, приемник интерпретирует состояние сигнала как новую порцию информации. Например, если такт равен 0,3 секунды, а сигнал имеет два состояния и число 1 кодируется потенциалом 5 вольт, то присутствие на входе приемника сигнала величиной 5 вольт в течение 3 секунд означает получение информации, представленной двоичным числом 1111111111.

Скорость изменения информационного сигнала равна количеству тактов изменения информационного параметра несущего периодического сигнала в секунду. Она измеряется в **бодах**.

Например, если такт передачи информации равен 0,1 секунды, то сигнал изменяется со скоростью 10 бод.

Скорость передачи информации, которая в данном случае является *пропускной способностью*, измеряемая в *битах в секунду*, может быть как выше, так и ниже скорости изменения информационного сигнала, измеряемой в *бодах*. Это соотношение зависит от числа состояний информационного параметра. Например, если он имеет более двух различимых состояний, то при равных тактах и соответствующем методе кодирования информационная скорость в битах в секунду может быть *выше*, чем скорость изменения информационного сигнала в бодах.

Пусть информационными параметрами являются фаза и амплитуда синусоиды, причем различаются 4 состояния фазы в 0°, 90°, 180° и 270° и два значения амплитуды сигнала — тогда информационный сигнал может иметь 8 различимых состояний. Это означает, что любое состояние этого сигнала несет 3 бита информации — 000, 001,..., 111. В этом случае модем, работающий со скоростью 2400 бод (меняющий информационный сигнал 2400 раз в секунду), передает информацию со скоростью 7200 бит/с, так как при одном изменении сигнала передается 3 бита информации.

Если сигнал имеет два состояния (то есть несет информацию в 1 бит), то информационная скорость обычно совпадает с количеством бодов. Однако может наблюдаться и обратная картина, когда информационная скорость оказывается *ниже* скорости изменения информационного сигнала в бодах. Это происходит в тех случаях, когда для надежного распознавания приемником пользовательской информации каждый бит в последовательности кодируется несколькими изменениями информационного параметра несущего сигнала. Например, при кодировании единичного значения бита импульсом положительной полярности, а нулевого значения бита — импульсом отрицательной полярности физический сигнал дважды изменяет свое состояние при передаче каждого бита. При таком кодировании скорость линии в битах в секунду в два раза ниже, чем в бодах.

Из приведенных примеров видно, что пропускная способность линии связи тем выше, чем короче такт (выше частота периодического несущего сигнала) и чем больше устойчивых распознаваемых состояний имеет информационный сигнал. Однако эта скорость не может расти неограниченно, поскольку с увеличением частоты периодического несущего сигнала увеличивается и ширина спектра этого сигнала. Линия связи передает этот спектр синусоид с теми искажениями, которые определяются ее полосой пропускания. Чем больше несоответствие между полосой пропускания линии и шириной спектра передаваемых информационных сигналов, тем больше сигналы искажаются и тем вероятнее ошибки в распознавании различных состояний сигнала принимающей стороной. Это ограничивает число состояний сигнала, а значит, уменьшает скорость передачи информации. Таким образом, мы приходим к поиску баланса между характеристиками среды передачи (зату-

хание, полоса пропускания и др.) и методом кодирования, чтобы добиться максимальной величины пропускной способности.

Подробно различные методы кодирования рассматриваются в следующей главе этой книги.

Соотношение полосы пропускания и пропускной способности

Связь между полосой пропускания линии и ее пропускной способностью вне зависимости от принятого способа физического кодирования установил **Клод Шеннон**. Эта связь выражается следующим образом:

$$C = F \log_2 (1 + SNR) \text{ или } C = F \log_2 (1 + P_с/P_ш).$$

где C - пропускная способность линии в битах в секунду, F - ширина полосы пропускания линии в герцах, SNR - соотношение «сигнал/шум», $P_с$ - мощность сигнала, $P_ш$ - мощность шума.

Из представленного соотношения следует, что теоретического предела пропускной способности линии с фиксированной полосой пропускания не существует. Однако на практике такой предел имеется, поскольку повысить пропускную способность линии можно или за счет увеличения мощности передатчика или посредством уменьшения мощности шума в линии связи. Обе эти составляющие поддаются изменению с большим трудом. Повышение мощности передатчика ведет к значительному увеличению его габаритов и стоимости. Снижение уровня шума требует применения специальных кабелей с хорошими защитными экранами, что тоже весьма дорого, а также снижения шума в передатчике и промежуточной аппаратуре, чего непросто достичь технически. К тому же влияние мощностей полезного сигнала и шума на пропускную способность ограничено логарифмической зависимостью, которая растет далеко не так быстро, как прямо пропорциональная. Так, при достаточно типичном исходном значении отношения мощности сигнала к мощности шума, равном 100, повышение мощности передатчика в 2 раза позволит увеличить пропускную способность линии не более чем на 15 процентов.

Близким по сути к формуле Шеннона является другое соотношение, полученное *Найквистом*, которое также определяет *максимально возможную пропускную способность линии связи, но без учета шума в линии*:

$$C = 2F \log_2 M.$$

где M — количество различимых состояний информационного параметра.

Если сигнал имеет два различимых состояния, то максимально возможная пропускная способность равна удвоенному значению ширины полосы пропускания линии связи. Если же в передатчике используется более двух устойчивых состояний сигнала для кодирования данных, то максимально возможная пропускная способность линии повышается, так как за один такт работы передатчик передает несколько битов исходных данных. Например, если сигнал может принимать 4 значения электрического потенциала, то за один такт передается 2 бита данных, если 16 значений, то 4 бита, и т. д. Тем самым пропускная способность линии повышается в 2 или 4 раза при той же самой тактовой частоте передатчика.

Хотя в формуле Найквиста наличие шума в явном виде не учитывается, косвенно его влияние отражается в выборе количества состояний информационного сигнала. Для повышения

пропускной способности линии связи следовало бы увеличивать количество состояний, но, как отмечено, этому препятствует шум на линии. Например, мы не всегда можем увеличить пропускную способность линии за счет увеличения числа состояний, если амплитуда шума время от времени будет превышать разницу между соседними уровнями сигнала. Так что, возможно, вместо желательных 16 состояний нам придется ограничиться всего четырьмя, а то и двумя. Количество возможных состояний сигнала фактически ограничивается соотношением мощности сигнала и шума по формуле Шеннона, а формула Найквиста определяет предельную скорость передачи данных в том случае, когда количество состояний уже выбрано с учетом возможностей устойчивого распознавания сигнала приемником.

Проводные линии связи

Сегодня как для внутренней проводки (кабели зданий), так и для внешней (прокладываемой вне зданий) чаще всего применяются три класса проводных линий связи:

- экранированная и неэкранированная витая пара;
- коаксиальные кабели;
- волоконно-оптические кабели.

Экранированная и неэкранированная витая пара

Витой парой называется скрученная пара проводов. Этот вид среды передачи данных очень популярен и составляет основу большого количества как внутренних, так и внешних кабелей. Кабель может состоять из нескольких скрученных пар (внешние кабели иногда содержат до нескольких десятков таких пар).

Скручивание проводов снижает влияние внешних и взаимных помех на полезные сигналы, передаваемые по кабелю.

Основные особенности конструкции кабелей схематично показаны на рис. 6.19.

Кабели на основе витой пары являются *симметричными*, то есть они состоят из двух одинаковых в конструктивном отношении проводников. Симметричный кабель на основе витой пары может быть как *экранированным*, так и *неэкранированным*.

Нужно отличать *электрическую* изоляцию проводящих жил, которая имеется в любом кабеле, от *электромагнитной* изоляции. Первая состоит из непроводящего диэлектрического слоя — бумаги или полимера, например поливинилхлорида или полистирола. Во втором случае, помимо электрической изоляции, проводящие жилы помещаются также внутрь электромагнитного экрана, в качестве которого чаще всего применяется проводящая медная оплетка.

Экранированная витая пара (Shielded Twisted Pair, **STP**) хорошо защищает передаваемые сигналы от внешних помех, а также меньше излучает электромагнитные колебания вовне, что, в свою очередь, защищает пользователей сетей от вредного для здоровья излучения. Экранироваться может как кабель в целом, так и каждая отдельная пара для уменьшения перекрестных наводок. Наличие заземляемого экрана удорожает кабель и усложняет его прокладку, поэтому экранированную витую пару применяют только в тех случаях, когда это действительно необходимо, например, из-за больших внешних помех и/или повышенных требований к надежности передачи данных.

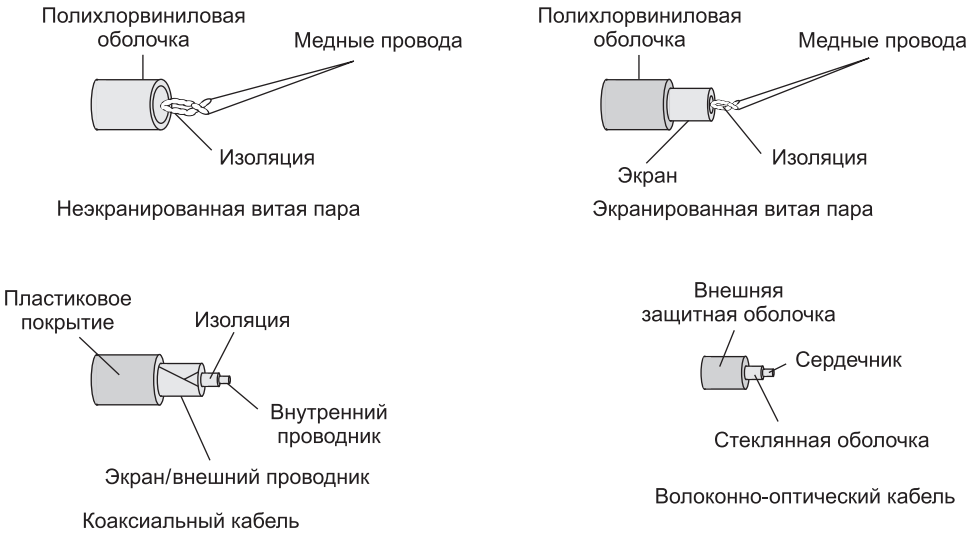


Рис. 6.19. Устройство кабелей

Неэкранированная витая пара (Unshielded Twisted Pair, **UTP**) обеспечивает защиту от внешних помех только за счет скручивания проводов в пары, что, естественно, не является такой эффективной мерой, как экранирование, но во многих случаях оказывается достаточным для обеспечения передачи данных с нужным качеством.

Кабели на основе витой пары, используемые для проводки внутри здания, разделяются в международных стандартах на *категории* (от 1 до 7).

Параметры кабелей категорий 1–6 определяются стандартами TIA/EIA-568 (разработаны организацией Telecommunication Industry Association, бывшей долгое время подразделением ныне упраздненной организации Electronic Industries Alliance, — от названий этих организаций и происходит аббревиатура TIA/EIA), а также близкими к ним стандартами ISO/IEC 11801 (последние также определяют кабели категории 7 и 8).

Все кабели на витой паре (независимо от категории) выпускаются в четырехпарном исполнении. Каждая из четырех пар кабеля имеет определенные цвет и шаг скрутки.

Кабели категорий 1, 2, 3 и 4 сегодня практически не применяются.

Кабели *категории* 5 были специально разработаны для поддержки высокоскоростных протоколов. Их характеристики определяются в диапазоне до 100 МГц. Существует улучшенная версия категории 5e (5 **e**nhanced), которая была разработана специально для более качественной поддержки протокола Gigabit Ethernet — в основном, за счет более жестких ограничений на перекрестные наводки.

Появление технологии 10G Ethernet привело к стандартизации более качественных кабелей *категорий* 6, 6*a*, 7 и 8. Для кабеля категории 6 характеристики определяются до частоты 250 МГц, категории 6*a* — до 500 МГц, категории 7 — до 600 МГц. Кабели категории 7 обязательно экранируются, причем как каждая пара, так и весь кабель в целом. Кабель категории 6 может быть как экранированным, так и неэкранированным. Максимальная длина сегмента 10G Ethernet на кабеле категории 6 равна 55 м, а на кабелях категорий 6*a*

и 7 — 100 м. Для кабелей категории 8 характеристики определяются до частоты 2000 МГц и расстояния кабеля до 30 м.

Коаксиальный кабель

Коаксиальный кабель состоит из несимметричных пар проводников. Каждая пара представляет собой внутреннюю медную жилу и соосную с ней внешнюю жилу, которая может быть полой медной трубой или оплеткой, отделенной от внутренней жилы диэлектрической изоляцией. Внешняя жила играет двоякую роль — по ней передаются информационные сигналы, и она же является экраном, защищающим внутреннюю жилу от внешних электромагнитных полей. Существует несколько типов коаксиального кабеля, отличающихся характеристиками и областями применения: для локальных компьютерных сетей, для глобальных телекоммуникационных сетей, для кабельного телевидения и т. п.

Согласно современным стандартам, коаксиальный кабель не считается хорошим выбором при построении структурированной кабельной системы зданий. Далее приводятся основные типы и характеристики этих кабелей.

- **«Толстый» коаксиальный кабель** разработан для сетей Ethernet 10Base-5 с волновым сопротивлением 50 Ом и внешним диаметром около 12 мм. Этот кабель имеет достаточно толстый внутренний проводник диаметром 2,17 мм, который обеспечивает хорошие механические и электрические характеристики (затухание на частоте 10 МГц — не хуже 18 дБ/км). Однако этот кабель сложно монтировать — он плохо гнется.
- **«Тонкий» коаксиальный кабель** предназначен для сетей Ethernet 10Base-2. Обладая внешним диаметром около 5 мм и тонким внутренним проводником 0,89 мм, этот кабель не так прочен, как «толстый» коаксиал, зато обладает гораздо большей гибкостью, что удобно при монтаже. «Тонкий» коаксиальный кабель также имеет волновое сопротивление 50 Ом, но его механические и электрические характеристики хуже, чем у «толстого» коаксиального кабеля. Затухание в этом типе кабеля выше, чем в «толстом» коаксиальном кабеле, что приводит к необходимости уменьшать длину кабеля для получения одинакового затухания в сегменте.
- **Телевизионный кабель** с волновым сопротивлением 75 Ом широко применяется в кабельном телевидении. Существуют стандарты локальных сетей, позволяющие использовать такой кабель для передачи данных.
- **Твинаксиальный кабель** по конструкции похож на коаксиальный кабель, но отличается наличием двух внутренних проводников. Такой кабель применяется в новых высокоскоростных стандартах 10G и 100G Ethernet для передачи данных на небольшие расстояния, распараллеливание потоков данных между двумя проводниками упрощает достижение высокой суммарной скорости.

Волоконно-оптический кабель

Волоконно-оптический кабель состоит из тонких (5–60 микрон) гибких стеклянных волокон (*волоконных световодов*), по которым распространяются световые сигналы. Это наиболее качественный тип кабеля — он обеспечивает передачу данных с очень высокой скоростью (до 100 Гбит/с и выше) на большие расстояния (80–100 км без промежуточного усиления), к тому же он лучше других типов передающей среды обеспечивает защиту

данных от внешних помех (в силу особенностей распространения света такие сигналы легко экранировать).

Каждый световод состоит из центрального проводника света (*сердечника,* или *сердцевины*) — стеклянного волокна, и стеклянной оболочки, обладающей меньшим показателем преломления, чем сердцевина. Распространяясь по сердцевине, лучи света не выходят за ее пределы, отражаясь от покрывающего слоя оболочки, так как она имеет более низкий коэффициент преломления. В зависимости от распределения показателя преломления и величины диаметра сердечника различают:

- многомодовое волокно со ступенчатым изменением показателя преломления (рис. 6.20, *а*);
- многомодовое волокно с плавным изменением показателя преломления (рис. 6.20, *б*);
- одномодовое волокно (рис. 6.20, *в*).

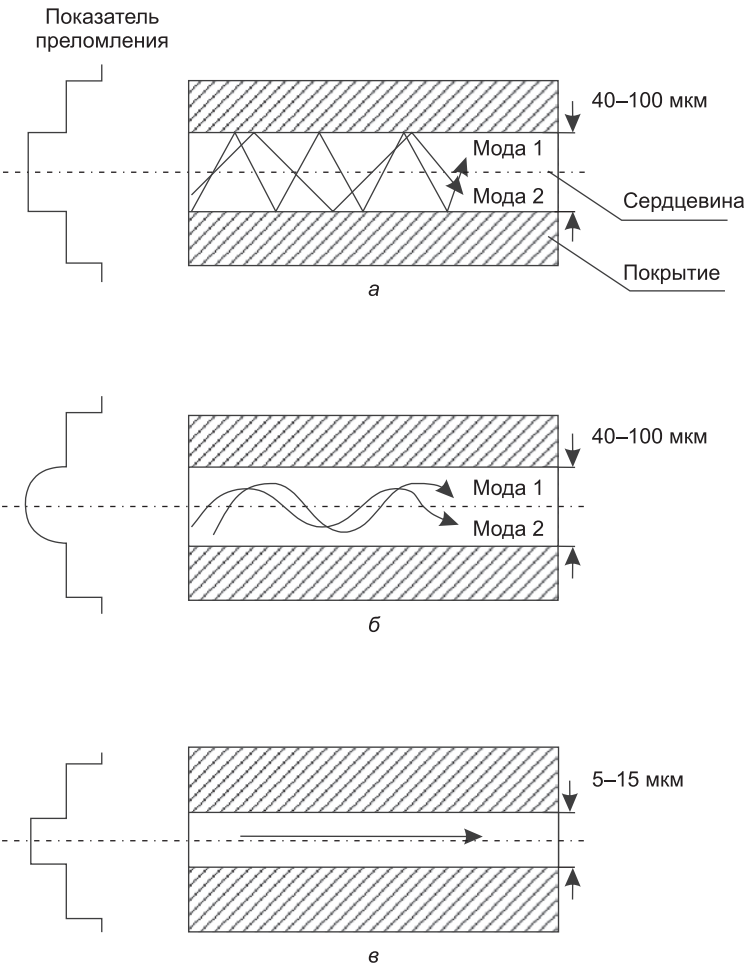


Рис. 6.20. Типы оптического кабеля

Понятие «мода» описывает режим распространения световых лучей в сердцевине кабеля.

В **одномодовом кабеле** (Single Mode Fiber, SMF) используется центральный проводник очень малого диаметра, соизмеримого с длиной волны света, — от 5 до 10 мкм. При этом практически все лучи света распространяются вдоль оптической оси световода, не отражаясь от внешнего проводника. Изготовление сверхтонких качественных волокон для одномодового кабеля представляет собой сложный технологический процесс, что делает одномодовый кабель достаточно дорогим. Кроме того, в волокно такого маленького диаметра достаточно сложно направить пучок света, не потеряв при этом значительную часть его энергии. Одномодовый кабель обладает очень *низким затуханием* — примерно –0,2 дБ/км для окна прозрачности волны размером в 1550 нм.

В **многомодовых кабелях** (Multi Mode Fiber, MMF) используются более широкие внутренние сердечники, которые легче изготовить технологически. В многомодовых кабелях во внутреннем проводнике одновременно распространяется несколько световых лучей, отражающихся от внешнего проводника под разными углами. Угол отражения луча называется **модой** луча. В многомодовых кабелях с плавным изменением коэффициента преломления режим отражения лучей имеет сложный характер. Возникающая при этом интерференция ухудшает качество передаваемого сигнала, что приводит к искажениям передаваемых импульсов. По этой причине технические характеристики многомодовых кабелей хуже, чем одномодовых.

Учитывая это, многомодовые кабели применяют в основном для передачи данных на скоростях не более 10 Гбит/с на небольшие расстояния (до 300–2000 м), а одномодовые — для передачи данных со сверхвысокими скоростями до сотен гигабит в секунду (при использовании технологии DWDM — до нескольких терабит в секунду) на расстояния до нескольких десятков и даже сотен километров (дальняя связь).

В качестве источников света в волоконно-оптических кабелях применяются:

- светодиоды, или светоизлучающие диоды (Light Emitted Diode, LED);
- полупроводниковые лазеры, или лазерные диоды.

Для одномодовых кабелей применяются только лазерные диоды, так как при столь малом диаметре оптического волокна световой поток, создаваемый светодиодом, невозможно без больших потерь направить в волокно — он имеет чересчур широкую диаграмму направленности излучения, в то время как лазерный диод — узкую. Более дешевые светодиодные излучатели используются только для многомодовых кабелей.

Стоимость волоконно-оптических кабелей ненамного превышает стоимость кабелей на витой паре, но проведение монтажных работ с оптоволокном обходится намного дороже из-за трудоемкости операций и высокой стоимости применяемого монтажного оборудования.

Понятно, что несмотря на отличные характеристики передачи световых сигналов, волоконно-оптические кабели не являются идеальными средами и вносят искажения в передаваемый сигнал.

Искажения сигнала в волоконно-оптических кабелях носят как линейный, так и нелинейный характер (линейность в данном случае определяется по отношению к интенсивности светового сигнала). К линейным искажениям относятся:

- **Затухание оптического сигнала**. Мощность сигнала уменьшается из-за поглощения света материалом волокна и примесями, рассеивания света из-за неоднородности

плотности волокна, а также из-за кабельных искажений, обусловленных деформацией волокон при прикладке кабеля. Затухание измеряется в децибелах на километр, имеет типичные значения от –0,2 до –0,3 (диапазон 1550 нм), от –0,4 до –1 (диапазон 1310 нм) и от –2 до –3 (диапазон 880 нм).

- **Хроматическая дисперсия** (Chromatic Dispersion, **CD**). Сигнал искажается из-за того, что волны различной длины распространяются вдоль волокна с различной скоростью. Так как прямоугольный импульс имеет спектр ненулевой ширины, из-за хроматической дисперсии составляющие его волны приходят на выход волокна с различной задержкой, и фронты импульса оказываются размытыми. Две составляющие вносят свой вклад в хроматическую дисперсию: **материальная**, отражающая зависимость коэффициента преломления материала сердечника от длины волны, и **волноводная**, вызванная различным поведением волн различной длины на границе между сердечником и оболочкой, то есть там, где изменяется коэффициент преломления. Хроматическая дисперсия оценивается отношением разницы времени распространения двух волн (в пикосекундах) в волокне определенной длины, обычно 1 км, к разнице длин волн (в нанометрах), то есть в пс/нм × км. Материальная составляющая хроматической дисперсии является положительной для волн в окне прозрачности 880 нм (то есть в этом диапазоне длинные волны распространяются быстрее), и отрицательной — для волн в окне прозрачности 1550 нм. В окне 1310 нм волны имеют дисперсию, близкую к нулю, при этом ноль достигается непосредственно в окрестности волны 1310 нм (такая волна называется длиной волны нулевой дисперсии λ_0 данного кабеля). Типичные значения хроматической дисперсии для окна 1310 нм не превышают 3–5 пс/нм × км, а для окна 1550 нм — 20–25 пс/нм × км.
- **Поляризационная модовая дисперсия** (Polarization Mode Dispersion, **PMD**). Световая мода имеет две взаимно перпендикулярные поляризационные составляющие. В волноводе с идеальным поперечным сечением, то есть представляющим собой окружность, эти составляющие распространяются с одинаковой скоростью. Так как реальные волноводы всегда имеют некоторую овальность, то скорости составляющих отличаются, что приводит к поляризационной дисперсии. Этот вид дисперсии растет пропорционально квадратному корню длины кабеля, поэтому измеряется в пикосекундах, отнесенных к квадратному корню длины кабеля (пс/$\sqrt{}$км). Типичные значения **PMD** лежат в диапазоне 0,1–0,5 пс/$\sqrt{}$км. Поляризационная дисперсия вносит меньший вклад в искажения оптических импульсов, чем хроматическая, но ее вклад возрастает при увеличении частоты модуляции сигнала.
- **Нелинейные искажения** имеют различную природу, они обусловлены зависимостью коэффициента преломления среды от интенсивности света (эффект Керра), а также эффектами рассеяния света в оптическом волокне (рассеяние Рамана, рассеяние Бриллюэна). Нелинейная зависимость таких эффектов от интенсивности светового потока затрудняет их компенсацию. На практике это проявляется в ограничении длины секций волоконно-оптических сетей без преобразования оптического сигнала в электрический и обратно (такая операция называется регенерацией оптического сигнала).

Стандарты волоконно-оптических кабелей разрабатываются в ITU-T. Рекомендации G.652, G.653, G.655 описывают характеристики одномодовых, а G.651.1 — многомодовых волоконно-оптических кабелей.

Структурированная кабельная система зданий

Структурированная кабельная система (Structured Cabling System) здания — это набор коммутационных элементов (кабелей, разъемов, коннекторов, кроссовых панелей и шкафов), а также методика их совместного использования, которая позволяет создавать регулярные легко расширяемые структуры связей в вычислительных сетях.

Структурированная кабельная система здания — это своего рода «конструктор», с помощью которого проектировщик сети строит нужную ему конфигурацию из стандартных кабелей, соединенных стандартными разъемами и коммутируемых на стандартных кроссовых панелях.

Здание само по себе представляет собой достаточно регулярную структуру — оно состоит из этажей, а каждый этаж, в свою очередь, состоит из определенного количества комнат, соединенных коридорами. Структура здания предопределяет структуру его кабельной системы (рис. 6.21).

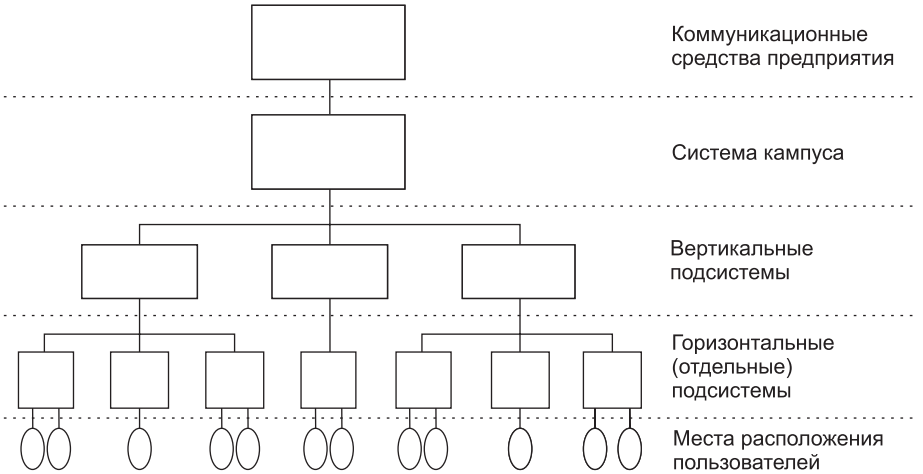


Рис. 6.21. Иерархия структурированной кабельной системы

Типичная структурированная кабельная система строится по иерархическому принципу, включая:

- *горизонтальные подсистемы*, соответствующие этажам здания, — они соединяют кроссовые шкафы этажа с розетками пользователей;
- *вертикальные подсистемы*, соединяющие кроссовые шкафы каждого этажа с центральной аппаратной здания;
- *подсистему кампуса* (иного комплекса зданий), объединяющую несколько зданий с главной аппаратной всего кампуса (эта часть кабельной системы обычно называется магистралью).

Использование структурированной кабельной системы вместо хаотически проложенных кабелей дает ее владельцу много преимуществ — при продуманной организации она может

стать *универсальной средой* передачи компьютерных данных в локальной вычислительной сети, организации локальной телефонной сети, передачи видеоинформации и даже передачи сигналов от датчиков пожарной безопасности или охранных систем. Подобная универсализация позволяет автоматизировать многие процессы корпоративного контроля, мониторинга, управления корпоративными хозяйственными службами и системами жизнеобеспечения.

Кроме того, применение структурированной кабельной системы делает *более экономичным* добавление новых пользователей и изменение их мест размещения. Известно, что стоимость кабельной системы определяется в основном не стоимостью кабеля, а стоимостью работ по его прокладке. Поэтому выгоднее провести однократную работу по прокладке кабеля, возможно, с большим запасом по длине, чем несколько раз выполнять прокладку с целью наращивания длины кабеля.

ГЛАВА 7 Кодирование и мультиплексирование данных

Виды кодирования

В предыдущей главе мы рассмотрели основные понятия кодирования информации при передаче через линии связи. Теперь изучим подробнее наиболее распространенные способы кодирования.

Начнем с краткой *классификации* видов кодирования. Она учитывает два фактора: характер передаваемой информации и характер сигналов, используемых для кодирования.

Информация может быть дискретной (двоичные данные компьютеров) или аналоговой (звуковые колебания, электромагнитные колебания, интенсивность света). Точно так же и сигналы, используемые для кодирования исходной информации, могут иметь дискретную (например, электрические сигналы с несколькими значениями потенциала) или аналоговую (электрические сигналы, радиоволны, световые волны) природу.

В результате все типы кодирования сводятся к следующим (рис. 7.1):

1) кодирование дискретной информации с помощью дискретных сигналов (Д-Д);
2) кодирование дискретной информации с помощью аналоговых сигналов (Д-А);

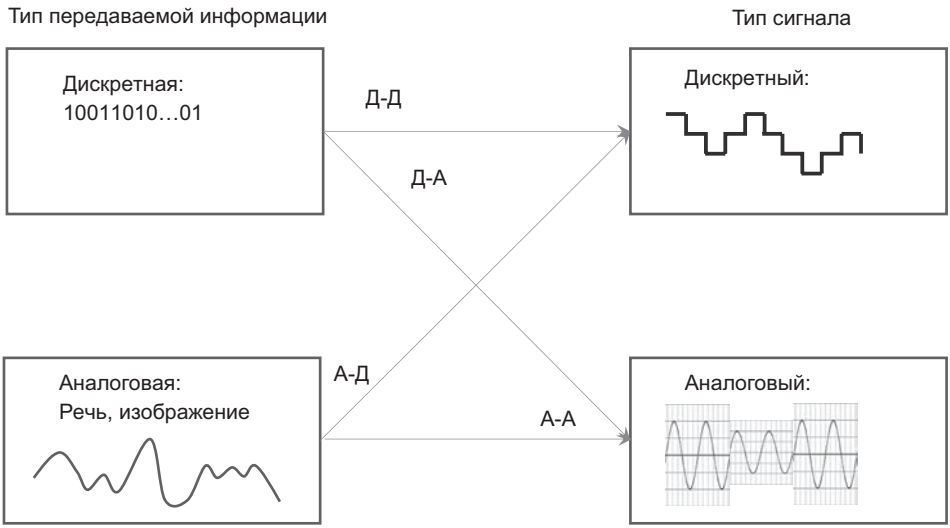


Рис. 7.1. Типы информации и сигналов при кодировании

3) кодирование аналоговой информации с помощью аналоговых сигналов (А-А);

4) кодирование аналоговой информации с помощью дискретных сигналов (А-Д).

В том случае, когда для представления информации на линии связи используются аналоговые сигналы, применяют не только термин «кодирование», но и термин «модуляция». Эти термины можно считать синонимами. Условимся, что в дальнейшем мы будем использовать термин «кодирование» как общий для любого типа сигналов, а термин «модуляция» — как частный случай кодирования, когда нужно подчеркнуть аналоговый характер применяемых для кодирования сигналов.

Кодирование дискретной информации

Этапы кодирования

Основные составляющие процесса кодирования дискретной информации показаны на рис. 7.2.

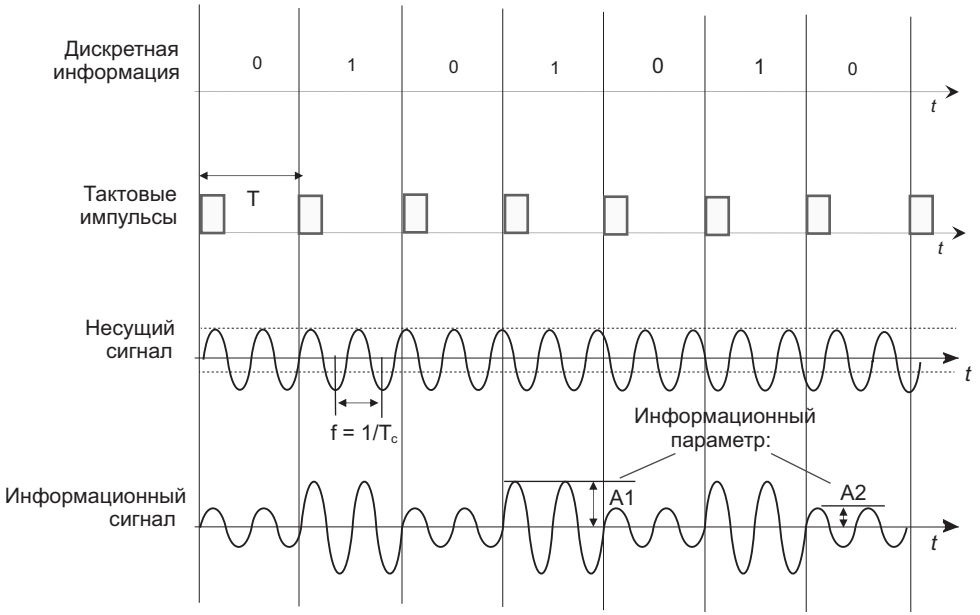


Рис. 7.2. Основные составляющие процесса кодирования

На вход передатчика поступает *дискретная информация*, то есть последовательность нулей и единиц. Примером передатчика может быть сетевой адаптер компьютера, на который через внутреннюю шину компьютера поступают данные, выработанные некоторой программой.

Передатчик работает с определенной *тактовой частотой*, посылая приемнику *тактовые импульсы* с периодом *T*. В нашем примере в каждом такте передается один бит инфор-

мации. Это подразумевает, что выбранный *информационный сигнал* имеет два значения информационного параметра.

Передатчик генерирует *несущий сигнал*, который в нашем примере является синусоидальным электрическим сигналом с частотой f и периодом T_c.

В каждом такте передатчик изменяет (модулирует) несущий сигнал в соответствии со значением бита информации, передаваемого на этом такте. *Информационным параметром* в нашем примере является амплитуда синусоидального сигнала, которая может принимать два значения — A1 и A2. Значением A1 кодируется ноль, а A2 соответствует единице.

Такт несущей частоты T_c выбирается таким образом, чтобы информационный параметр сигнала (в данном случае его амплитуда) мог быть устойчиво распознан приемником. В нашем примере он в два раза меньше такта передачи информации.

Модулированный информационный сигнал передается по линии связи и принимается приемником. Хотя в приведенной схеме используется кодирование аналоговым сигналом, ее основные элементы справедливы и для кодирования дискретными сигналами.

Спектр информационного сигнала

Для успешного распознавания приемником дискретной информации, переданной передатчиком, нужно, чтобы спектр информационного сигнала укладывался в полосу пропускания линии связи. По этой причине ширина спектра информационного сигнала и его положение на частотной оси являются одними из главных факторов, которые принимаются во внимание при создании некоторого метода кодирования.

Спектр результирующего информационного сигнала зависит от *типа кодирования* и *частоты изменения информационного сигнала* (напомним, что она измеряется в бодах). Эта частота является тактовой частотой передатчика.

Сравним спектры дискретного и аналогового кодирования с одинаковой тактовой частотой и посмотрим, какие требования предъявляют эти типы кодирования к полосе пропускания канала.

Рассмотрим сначала *спектр информационного сигнала при дискретном кодировании*. В качестве типичного представителя этого типа кодов возьмем *кодирование потенциалом электрического сигнала с двумя состояниями*. Пусть логическая единица кодируется положительным потенциалом, а логический ноль — отрицательным потенциалом такой же величины. Для упрощения вычислений предположим, что передается информация, состоящая из бесконечной последовательности чередующихся единиц и нулей. Результирующий информационный сигнал имеет вид последовательности прямоугольных импульсов чередующейся полярности.

Для такого вида сигнала спектр может быть получен аналитически из формул Фурье для периодической функции. Пусть дискретные данные передаются с тактовой частотой N бод. Так как потенциальный код имеет два состояния сигнала, то и битовая скорость передачи данных будет равна N бит/с. Спектр такого кода состоит из постоянной составляющей нулевой частоты и бесконечного ряда гармоник с частотами f_0, $3f_0$, $5f_0$, $7f_0$, ..., где $f_0 = N/2$. Амплитуды этих гармоник убывают достаточно *медленно* — с коэффициентами 1/3, 1/5, 1/7, ... от амплитуды гармоники f_0 (рис. 7.3, *а*). Это означает, что *для качественной передачи потенциального кода нужен канал с относительно широкой полосой пропускания*. Например, для передачи данных, закодированных таким способом, со скоростью 100 Мбит/с канал

должен иметь полосу пропускания от 50 МГц до 350 МГц, если считать, что основной вклад в форму сигнала вносят только первые четыре гармоники.

На рисунке показана полоса пропускания некоторого канала, которая хорошо пропускает только первые две гармоники потенциального сигнала, из-за чего сигнал на выходе будет сильно искажен. Следовательно, данный канал не подходит для выбранного типа кодирования и тактовой частоты. Заметим, что ширина спектра потенциального сигнала ($N/2$, $7N/2$), равная $3N$, прямо пропорциональна тактовой частоте передатчика N. В отсутствие возможности замены канала связи допустимо уменьшить тактовую частоту передатчика, что позволит добиться устойчивой передачи данных с меньшей скоростью. Для рассматриваемого примера уменьшение тактовой частоты в три раза приведет к желаемому результату, так как тогда первые четыре гармоники будут укладываться в полосу пропускания канала.

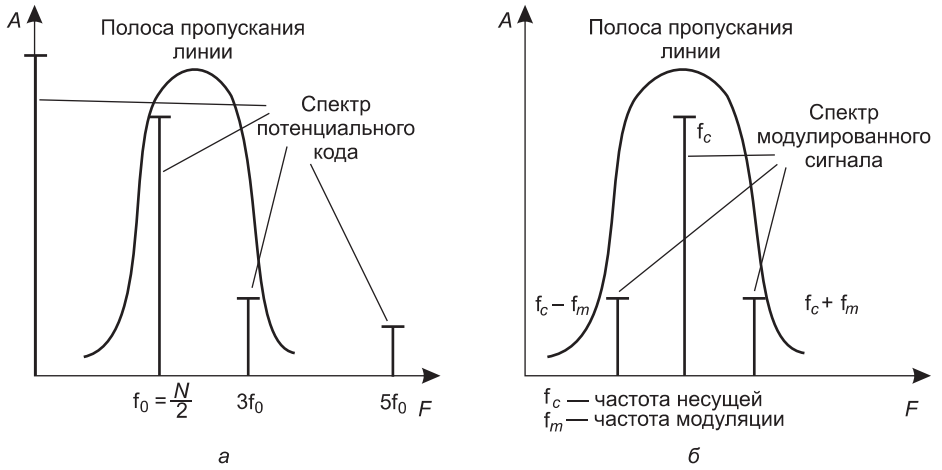


Рис. 7.3. Спектры сигналов при потенциальном кодировании и амплитудной модуляции

При анализе спектра потенциального сигнала мы предположили, что передается чередующаяся последовательность единиц и нулей. Это упрощение позволило нам найти аналитическую формулу для всех гармоник спектра. Но на практике такая ситуация наблюдается не всегда, а значит, и спектр сигнала будет другим. Например, передача длинной последовательности нулей или единиц сдвигает спектр в сторону низких частот, а в особом случае, когда передаваемые данные состоят только из единиц (или только из нулей), спектр состоит из гармоники нулевой частоты. При передаче чередующихся единиц и нулей постоянная составляющая отсутствует. Поэтому более адекватным является утверждение о том, что спектр потенциального сигнала при передаче произвольных данных занимает полосу от некоторой величины, близкой к нулю, до примерно $7f_0$. Гармониками с частотами выше$7f_0$ можно пренебречь из-за их малого вклада в результирующий сигнал. То есть спектр потенциального кода имеет границы (0, $7N/2$), включающие низкие частоты. Это обстоятельство делает его малопригодным для передачи компьютерных данных по абонентским телефонным каналам, для которых характерна полоса пропускания (300 Гц, 3400 Гц).

Теперь рассмотрим *спектр информационного сигнала в случае аналогового кодирования*. В качестве типового представителя этого класса кодирования возьмем амплитудную модуляцию синусоидальной несущей. Вид результирующего информационного сигнала показан

на рис. 7.3, *б*. Спектр сигнала такого вида может быть представлен гармоникой частоты f_c и двумя боковыми гармониками ($f_c + f_m$) и ($f_c - f_m$), где f_m — частота изменения информационного параметра синусоиды, то есть тактовая частота передатчика f_m = N. Мощность гармоник более высоких частот убывает быстро, в связи с чем ими можно пренебречь. Отсюда следует, что амплитудная аналоговая модуляция имеет спектр сигнала шириной $2N$, что у́же, чем спектр потенциального кода с той же тактовой частотой, поскольку $2N < 7N/2$. Например, при тактовой частоте 10 Мгц потенциальный код будет иметь спектр шириной в 35 МГц, а амплитудно-модулированный сигнал — только 20 МГц.

Спектр амплитудно-модулированного сигнала не только у́же спектра потенциального кода, но и может быть перемещен по оси частот в зону полосы пропускания линии связи путем варьирования частотой f_c несущего сигнала. Это делает амплитудную модуляцию более подходящим способом передачи дискретных данных по телефонным каналам.

Выбор способа кодирования

При выборе способа кодирования нужно одновременно стремиться к достижению нескольких целей:

- минимизировать ширину спектра сигнала, полученного в результате кодирования;
- обеспечивать синхронизацию между передатчиком и приемником;
- обеспечивать устойчивость к шумам;
- обнаруживать и по возможности исправлять битовые ошибки;
- минимизировать мощность передатчика.

Более *узкий спектр сигнала* позволяет на одной и той же линии (с одной и той же полосой пропускания) добиваться более высокой скорости передачи данных.

Как мы видели ранее, спектр сигнала при некотором выбранном методе кодирования пропорционально увеличивается при увеличении тактовой частоты передатчика, например, для потенциального кодирования эта зависимость прямо пропорциональна. Поэтому, зафиксировав способ кодирования, мы можем повышать тактовую частоту передатчика и, следовательно, битовую скорость передаваемых дискретных данных до некоторого предела, до тех пор, пока спектр сигнала еще помещается в полосу пропускания линии. Более высокой битовой скорости мы при данном методе кодирования достичь не сможем, так как при дальнейшем повышении тактовой частоты передатчика боковые составляющие спектра будут обрезаться линией и сигналы начнут приходить на приемник искаженными, так что приемник не сможет надежно распознавать биты передаваемой информации.

Но если применить другой метод кодирования, который при той же тактовой частоте приводит к сигналам более узкого спектра, то, очевидно, удастся повысить тактовую частоту до более высокого предела. Если новый и старый методы кодирования использовали одно и то же число состояний сигнала, то мы добьемся выигрыша в битовой скорости — во столько раз, во сколько при одной и той же частоте спектр нового метода кодирования уже старого.

Синхронизация передатчика и приемника нужна для того, чтобы приемник точно знал, в какой момент времени считывать новую порцию информации с линии связи. При передаче дискретной информации время всегда разбивается на такты одинаковой длительности. Приемник старается считать новый сигнал в середине каждого такта, синхронизируя таким образом свои действия с передатчиком.

Проблему синхронизации устройств, связанных сетью, решить сложнее, чем синхронизацию устройств, близко расположенных друг к другу, например блоков внутри компьютера. На небольших расстояниях хорошо работает схема, основанная на отдельной *тактирующей линии связи*, так что информация снимается с линии данных только в момент прихода тактового импульса. В сетях использование этой схемы вызывает трудности из-за неоднородности характеристик проводников в кабелях. На больших расстояниях неравномерность скорости распространения сигнала может привести к тому, что тактовый импульс придет настолько позже или раньше соответствующего сигнала данных, что бит данных будет пропущен или, наоборот, считан повторно. Другой причиной, по которой в сетях отказываются от использования тактирующих импульсов, является экономия проводников в дорогостоящих кабелях.

В сетях для решения проблемы синхронизации применяются так называемые **самосинхронизирующиеся коды**, сигналы которых несут для приемника указания о том, в какой момент времени начать распознавание очередного бита (или нескольких битов, если код ориентирован более чем на два состояния сигнала). Любой резкий перепад сигнала — **фронт** — может служить указанием на необходимость синхронизации приемника с передатчиком. При использовании синусоид в качестве несущего сигнала результирующий код обладает свойством самосинхронизации, так как изменение амплитуды несущей частоты дает возможность приемнику определить момент очередного такта.

Распознавание и коррекцию искаженных данных сложно осуществить средствами физического уровня, поэтому чаще всего эту работу берут на себя вышележащие протоколы: канальный, сетевой, транспортный или прикладной. В то же время распознавание ошибок на физическом уровне экономит время, так как приемник не ждет полного помещения кадра в буфер, а отбраковывает его сразу при распознавании ошибочных битов внутри кадра.

Требования, предъявляемые к методам кодирования, являются взаимно противоречивыми, поэтому каждый из рассматриваемых далее популярных методов кодирования обладает своими индивидуальными достоинствами и недостатками в сравнении с другими.

Кодирование дискретной информации дискретными сигналами

Потенциальный код NRZ

Рисунок 7.4, *а* иллюстрирует уже упомянутый ранее метод *потенциального кодирования*, называемый также кодированием **без возвращения к нулю** (Non Return to Zero, **NRZ**). Последнее название отражает то обстоятельство, что, в отличие от других методов кодирования, при передаче последовательности единиц сигнал не возвращается к нулю в течение такта.

К достоинствам метода NRZ относятся:

- Простота реализации.
- Хорошая распознаваемость кода (благодаря наличию двух резко отличающихся потенциалов).
- Основная гармоника f_0 имеет достаточно низкую частоту (равную $N/2$ Гц, как было показано в предыдущем разделе), что приводит к относительно узкому спектру.

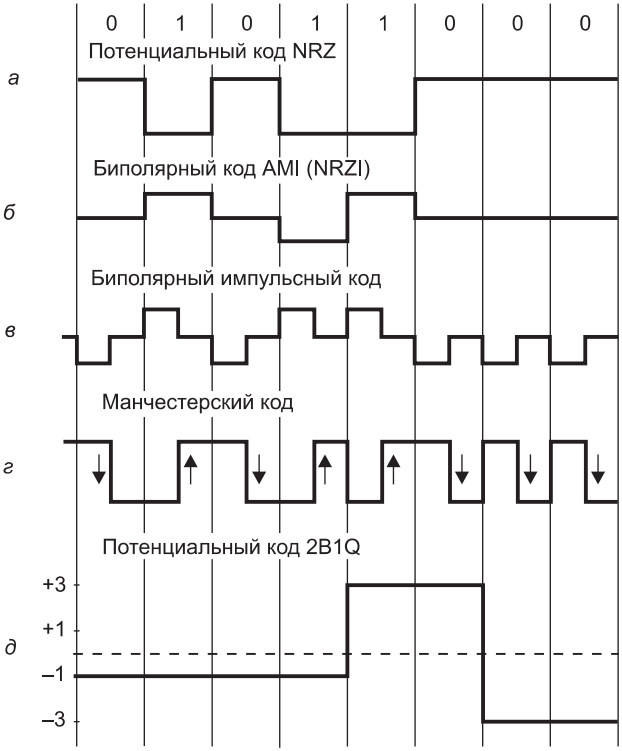


Рис. 7.4. Способы дискретного кодирования данных

Назовем и недостатки метода NRZ:

- Метод не обладает свойством самосинхронизации. Длинная последовательность единиц или нулей приводит к тому, что сигнал не изменяется в течение многих тактов, так что приемник не имеет возможности синхронизироваться с передатчиком.
- Из-за наличия низкочастотной составляющей, которая приближается к постоянному сигналу при передаче длинных последовательностей единиц или нулей линии связи, не обеспечивающие прямого гальванического соединения между приемником и источником, часто не поддерживают этот вид кодирования.

Кодирование NRZ применяется как для электрических, так и для световых сигналов, и в последнем случае сигналом является не уровень потенциала, а наличие или отсутствие света в оптическом волокне.

Биполярное кодирование AMI

Одной из модификаций метода NRZ является метод **биполярного кодирования с альтернативной инверсией** (Alternate Mark Inversion, **AMI**). В этом методе применяются три уровня потенциала — отрицательный, нулевой и положительный (см. рис. 7.4, *б*). Для кодирования логического нуля используется нулевой потенциал, а логическая единица

кодируется либо положительным потенциалом, либо отрицательным, при этом потенциал каждой новой единицы противоположен потенциалу предыдущей.

При передаче *длинных последовательностей единиц* код AMI частично решает проблемы наличия постоянной составляющей и отсутствия самосинхронизации, присущие коду NRZ. В этих случаях сигнал на линии представляет собой последовательность разнополярных импульсов с тем же спектром, что и у кода NRZ, передающего чередующиеся нули и единицы, то есть без постоянной составляющей и с основной гармоникой $N/2$ Гц (где N — битовая скорость передачи данных). *Длинные последовательности нулей* для кода AMI столь же опасны, как и для кода NRZ — сигнал вырождается в постоянный потенциал нулевой амплитуды.

Потенциальный код NRZI

Потенциальный код с инверсией при единице (Non Return to Zero with ones Inverted, **NRZI**) при передаче нуля сохраняет потенциал, который был установлен на предыдущем такте, а при передаче единицы инвертирует его на противоположный.

Код NRZI обладает лучшей самосинхронизацией, чем NRZ, так как при передаче единицы сигнал меняется. Тем не менее при передаче длинных последовательностей нулей сигнал не меняется (например, при передаче последних трех нулей на рис. 7.4, *а*) и, значит, у приемника исчезает возможность синхронизации с передатчиком на значительное время, что может приводить к ошибкам распознавания данных.

Биполярный импульсный код

Помимо *потенциальных кодов* в сетях используются *импульсные коды*, в которых данные представлены полным импульсом или же его частью — фронтом. Наиболее простым кодом такого рода является **биполярный импульсный код**, в котором единица представляется импульсом одной полярности, а ноль — другой (см. рис. 7.4, *в*). Каждый импульс длится половину такта. Подобный код обладает отличными самосинхронизирующими свойствами, но постоянная составляющая может присутствовать, например, при передаче длинной последовательности единиц или нулей. Кроме того, спектр у него шире, чем у потенциальных кодов. Так, при передаче всех нулей или единиц частота основной гармоники кода равна N Гц, что в два раза выше основной гармоники кода NRZ и в четыре раза выше основной гармоники кода AMI при передаче чередующихся единиц и нулей. Из-за слишком широкого спектра биполярный импульсный код используется редко.

Манчестерский код

В локальных сетях до недавнего времени самым распространенным был так называемый **манчестерский код** (см. рис. 7.4, *г*). Он применяется в технологии 10 Мбит/с Ethernet.

В манчестерском коде для кодирования единиц и нулей используется перепад потенциала, то есть фронт импульса. При манчестерском кодировании каждый такт делится на две части. Информация кодируется перепадами потенциала, происходящими в середине каждого такта. Единица кодируется перепадом от низкого уровня сигнала к высокому, а ноль — обратным перепадом. В начале каждого такта может происходить служебный перепад сигнала, если нужно представить несколько единиц или нулей подряд. Так как

сигнал изменяется по крайней мере один раз за такт передачи одного бита данных, то манчестерский код обладает хорошими самосинхронизирующими свойствами. При этом полоса пропускания манчестерского кода уже, чем у биполярного импульсного. Кроме того, у него нет постоянной составляющей, к тому же основная гармоника в худшем случае (при передаче последовательности единиц или нулей) имеет частоту N Гц, а в лучшем (при передаче чередующихся единиц и нулей) — $N/2$ Гц, как и у кодов AMI и NRZ. В среднем ширина полосы манчестерского кода в полтора раза уже, чем у биполярного импульсного кода, а основная гармоника колеблется вблизи значения $3N/4$. Манчестерский код имеет еще одно преимущество перед биполярным импульсным кодом: в последнем для передачи данных используются три уровня сигнала, а в манчестерском — два.

Избыточные коды

Избыточные коды основаны на разбиении исходной последовательности битов на порции, которые часто называют *символами*. Затем каждый исходный символ заменяется новым с бóльшим количество битов, чем исходный.

Например, в логическом коде **4B/5B**, используемом в технологии Fast Ethernet, исходные символы длиной 4 бит заменяются символами длиной 5 бит. Так как результирующие символы содержат избыточные биты, то общее количество битовых комбинаций в них больше, чем в исходных. Так, в коде 4B/5B результирующие символы могут содержать 32 битовые комбинации, в то время как исходные символы — только 16 (табл. 7.1). Поэтому в результирующем коде появляется возможность отобрать 16 таких комбинаций, которые не содержат большого количества нулей, а остальные посчитать **запрещенными кодами** (code violations). Помимо устранения постоянной составляющей и придания коду свойства самосинхронизации, избыточные коды позволяют приемнику распознавать искаженные биты. Если приемник принимает запрещенный код, значит, на линии произошло искажение сигнала.

Таблица 7.1. Соответствие исходных и результирующих кодов 4B/5B

Исходный код	Результирующий код	Исходный код	Результирующий код
0000	11110	1000	10010
0001	01001	1001	10011
0010	10100	1010	10110
0011	10101	1011	10111
0100	01010	1100	11010
0101	01011	1101	11011
0110	01110	1110	11100
0111	01111	1111	11101

После разбиения полученный код 4B/5B передается по линии путем преобразования с помощью какого-либо из методов потенциального кодирования, чувствительного только к длинным последовательностям нулей. Таким кодом является, например, NRZI. Символы кода 4B/5B длиной 5 бит гарантируют, что при любом их сочетании на линии не встретится более трех нулей подряд.

ПРИМЕЧАНИЕ

Буква B в названии кода 4B/5B означает, что элементарный сигнал имеет два состояния (от английского binary — двоичный). Имеются также коды и с тремя состояниями сигнала, например, в коде 8B/6T для кодирования 8 бит исходной информации используется код из 6 сигналов, каждый из которых имеет три состояния. Избыточность кода 8B/6T выше, чем кода 4B/5B, так как на 256 исходных кодов приходится $3^6 = 729$ результирующих символов.

Использование таблицы перекодировки не усложняет работу сетевых адаптеров и интерфейсных блоков коммутаторов и маршрутизаторов.

Для обеспечения заданной пропускной способности линии передатчик, использующий избыточный код, должен работать с повышенной тактовой частотой. Так, для передачи кодов 4B/5B со скоростью 100 Мбит/с требуется тактовая частота 125 МГц. При этом спектр сигнала на линии расширяется по сравнению со случаем, когда по линии передается не избыточный код. Тем не менее спектр избыточного потенциального кода оказывается у́же спектра манчестерского кода, что оправдывает дополнительный этап логического кодирования, а также работу приемника и передатчика на повышенной тактовой частоте. Чем ближе к единице соотношение числа исходных символов к общему числу символов, тем незначительнее становится повышение тактовой частоты передатчика. В скоростных версиях 10G Ethernet и 100G Ethernet применяется избыточный код 64B/66B.

Существует два метода, улучшающих код Bipolar AMI и основанных на искусственном искажении последовательности нулей запрещенными символами.

Рисунок 7.5 иллюстрирует использование метода **B8ZS** (Bipolar with 8-Zeros Substitution) и метода **HDB3** (High-Density Bipolar 3-Zeros) для корректировки кода AMI. Исходный код состоит из двух длинных последовательностей нулей: в первом случае — из восьми, а во втором — из пяти.

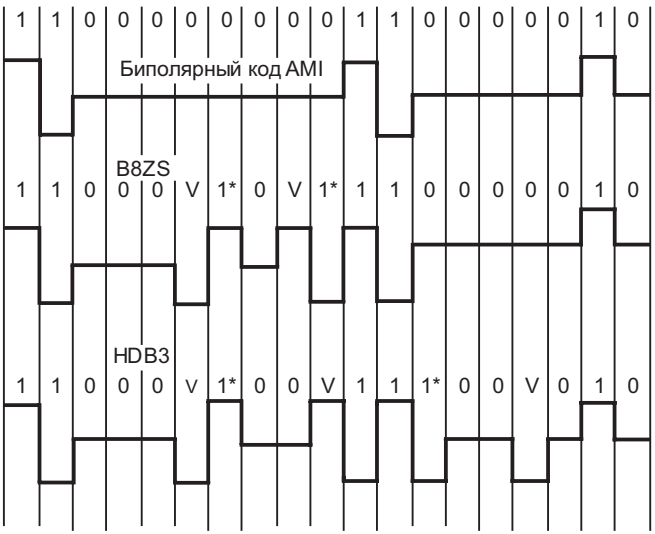


Рис. 7.5. Коды B8ZS и HDB3

Код B8ZS исправляет только последовательности, состоящие из восьми нулей. Для этого он после первых трех нулей вместо оставшихся пяти нулей вставляет пять сгруппированных символов: V-1*-0-V-1*. Здесь V обозначает сигнал единицы, запрещенной для данного такта полярности, то есть сигнал, не изменяющий полярность предыдущей единицы, 1* — сигнал единицы корректной полярности (знак звездочки отмечает тот факт, что в исходном коде в этом такте была не единица, а ноль). В результате на восьми тактах приемник наблюдает два искажения, при этом маловероятно, что это случилось из-за шума на линии или других сбоев передачи. Поэтому приемник считает такие нарушения кодировкой восьми последовательных нулей и после приема заменяет их исходными восемью нулями. Код B8ZS построен так, что его постоянная составляющая равна нулю при любых последовательностях двоичных цифр.

Код HDB3 исправляет любые четыре подряд идущих нуля в исходной последовательности. Правила формирования кода HDB3 более сложны, чем кода B8ZS. Каждые четыре нуля заменяются четырьмя сигналами, в которых имеется один сигнал V. Для подавления постоянной составляющей полярность сигнала V чередуется при последовательных заменах. Кроме того, для замены используются два образца четырехтактовых кодов. Если перед заменой исходный код содержал нечетное число единиц, задействуется последовательность 000V, а если число единиц было четным — последовательность 1*00V.

Улучшенные потенциальные коды обладают достаточно узкой полосой пропускания для любых последовательностей единиц и нулей, которые встречаются в передаваемых данных. На рис. 7.6 представлены спектры сигналов разных кодов, полученные при передаче произвольных данных, в которых различные сочетания нулей и единиц в исходном коде равновероятны. При построении графиков спектр усреднялся по всем возможным наборам исходных последовательностей. Естественно, что результирующие коды могут иметь

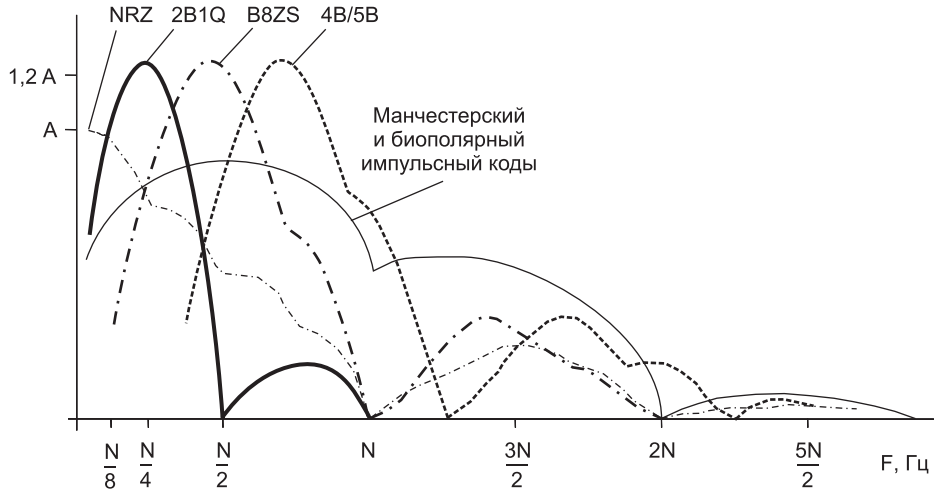


Рис. 7.6. Спектры потенциальных и импульсных кодов

и другое распределение нулей и единиц. Из рисунка видно, что потенциальный код NRZ обладает хорошим спектром с одним недостатком — у него имеется постоянная составляющая. Коды, полученные из потенциального путем логического кодирования, обладают более узким спектром, чем манчестерский, даже при повышенной тактовой частоте (на рисунке спектр кода 4B/5B должен был бы примерно совпадать с кодом B8ZS, но он сдвинут в область более высоких частот, так как его тактовая частота повышена на 1/4 по сравнению с другими кодами). Этим объясняется применение потенциальных избыточных и скремблированных кодов в современных технологиях, подобных FDDI, Fast Ethernet, Gigabit Ethernet, ISDN и т. п. вместо манчестерского и биполярного импульсного кодирования.

Избавиться от длинных последовательностей нулей в коде помогает **скремблирование** — «перемешивание» битов кода в соответствии с определенным алгоритмом, позволяющим приемнику выполнить обратное преобразование.

(S) *Скремблирование и компрессия данных*

Кодирование дискретной информации аналоговыми сигналами

При передаче дискретной информации посредством аналоговых сигналов единицы и нули кодируются изменением:

- амплитуды (как в примере на рис. 7.3);
- частоты;
- или фазы несущего синусоидального сигнала.

В случае, когда модулированные сигналы передают *дискретную* информацию, вместо термина «модуляция» иногда используется термин **манипуляция**: амплитудная манипуляция (Amplitude Shift Keying, ASK); частотная манипуляция (Frequency Shift Keying, FSK); фазовая манипуляция (Phase Shift Keying, PSK).

Пожалуй, самый известный пример применения модуляции при передаче дискретной информации — это передача компьютерных данных по телефонным каналам с помощью модема. Типичная амплитудно-частотная характеристика **канала тональной частоты** представлена на рис. 7.7. Этот составной канал проходит через коммутаторы телефонной сети и соединяет телефоны абонентов. Канал тональной частоты передает частоты в диапазоне от 300 до 3400 Гц и, таким образом, его полоса пропускания равна 3100 Гц. Хотя полоса пропускания тонального канала у́же спектра голоса, составляющего примерно 10 кГц, она достаточна для качественной передачи голоса (в чем мы убеждаемся, разговаривая по телефону). Однако, как показано ранее (см. раздел «Спектр информационного сигнала»), она не подходит для передачи компьютерных данных в виде прямоугольных импульсов с приемлемой битовой скоростью. Решение этой проблемы было найдено благодаря аналоговой модуляции. Устройство, которое выполняет функцию *модуляции* несущей синусоиды на передающей стороне и обратную функцию *демодуляции* на стороне приема, известно нам как **модем (мод**улятор-**дем**одулятор).

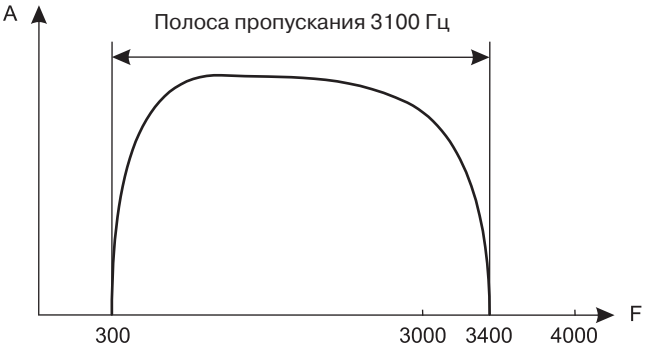


Рис. 7.7. Амплитудно-частотная характеристика канала тональной частоты

ПРИМЕЧАНИЕ

Как мы знаем, ширина спектра амплитудно-модулированного сигнала равна 2N, где N — тактовая частота передатчика, равная скорости передачи данных. Ширина спектра должна быть у́же полосы пропускания тонального канала, то есть 2N < 3100 Гц. Отсюда следует, что при амплитудной модуляции с двумя состояниями на телефонном канале скорость передатчика ограничена значением 1550 бит/с. Модемы, которые достигают более высоких скоростей, используют другие виды аналоговой модуляции.

На рис. 7.8 показаны различные типы модуляции, применяемые при передаче дискретной информации. Исходная последовательность битов передаваемой информации приведена на диаграмме, представленной на рис. 7.8, *а*.

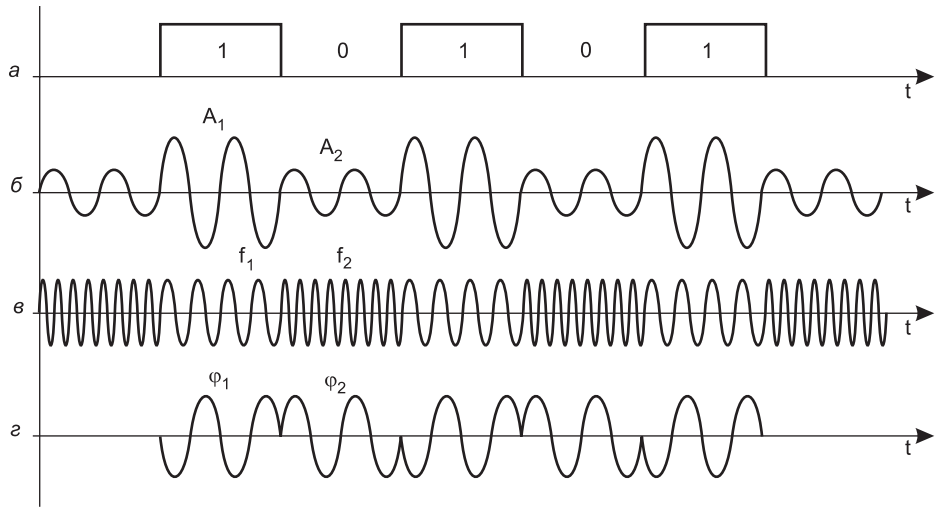


Рис. 7.8. Различные типы модуляции

При *амплитудной модуляции* для логической единицы выбирается один уровень амплитуды синусоиды несущей частоты, а для логического нуля — другой (рис. 7.8, *б*). Этот способ

редко используется в чистом виде на практике из-за низкой помехоустойчивости, но часто применяется в сочетании с другим видом модуляции — фазовой модуляцией.

При *частотной модуляции* значения нуля и единицы исходных данных передаются синусоидами с различной частотой — f_0 и f_1 (рис. 7.8, *в*). Этот способ модуляции не требует сложных схем и обычно применяется в низкоскоростных модемах, работающих на скоростях 300 и 1200 бит/с. При использовании только двух частот за один такт передается один бит информации, поэтому такой способ называется **двоичной частотной манипуляцией** (Binary FSK, BFSK). Могут также использоваться четыре различные частоты для кодирования двух битов информации в одном такте — такой способ носит название **четырехуровневой частотной манипуляции** (four-level FSK). Применяется также название **многоуровневая частотная манипуляция** (Multilevel FSK, MFSK).

При *фазовой модуляции* значениям двоичных данных соответствуют сигналы одинаковой частоты, но различной фазы, например, две различные фазы сигнала 0° и 180° соответствуют данным 0 и 1 (рис. 7.8, *г*), а при использовании четырех значений фазы 0°, 90°, 180° и 270° они соответствуют двоичным данным 00, 01, 10 и 11. В первом случае такая модуляция носит название **двоичной фазовой манипуляции** (Binary PSK, BPSK), во втором — **квадратурной фазовой манипуляции** (Quadrature PSK, QPSK).

Для повышения скорости передачи данных прибегают к *комбинированным методам модуляции*. Наиболее распространенными являются методы **квадратурной амплитудной модуляции** (Quadrature Amplitude Modulation, QAM). Они основаны на сочетании фазовой и амплитудной модуляции. На рис. 7.9 показан вариант квадратурной амплитудной модуляции, в котором используется 8 различных значений фазы и 4 значения амплитуды.

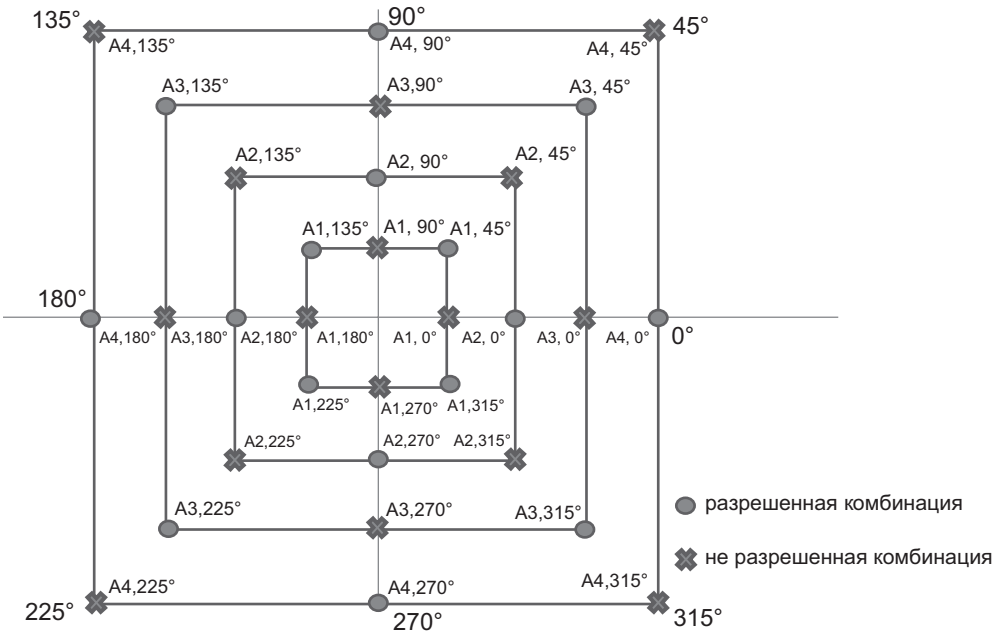


Рис. 7.9. Квадратурная амплитудная модуляция с 16 состояниями сигнала

Однако из 32 возможных комбинаций сигнала задействовано только 16, так как разрешенные значения амплитуд у соседних фаз должны отличаться. Например, если (A2, 0°) является разрешенной комбинацией, то (A2, 45°) — запрещена, так как 0° и 45° являются соседними фазами. Это повышает помехоустойчивость кода, но вдвое снижает скорость передачи данных. Каждая комбинация является одним из 16 состояний информационного параметра; за каждый такт передается 4 бита данных (0000 или 0001... или 1111).

Другим решением, повышающим надежность кода за счет введения избыточности, являются так называемые **решетчатые коды**. В них к каждым четырем битам информации добавляется пятый бит, который даже при наличии ошибок позволяет с большой степенью вероятности определить правильный набор четырех информационных битов.

Модуляторы, реализующие амплитудную, частотную и фазовую модуляцию, а также их комбинации (такие, как, например, 16-QAM) работают на основе математического представления синусоидального сигнала в виде двух составляющих — синфазной и квадратурной. Рассмотрим это представление для случая амплитудно-фазовой модуляции 16-QAM.

Модулированный сигнал можно представить в виде:

$$y(t) = a(t)\cos(2\pi f_н t - \varphi(t)),$$

где $a(t)$ — закон изменения амплитуды от времени, эта функция принимает одно из четырех значений амплитуды кода 16-QAM; $\varphi(t)$ — закон изменения фазы от времени, эта функция принимает одно из четырех разрешенных значений фазы кода 16-QAM, $f_н$ — частота несущей.

Преобразуем эту зависимость, используя тригонометрическое соотношение для косинуса суммы двух углов. Получим:

$$y(t) = a(t)\cos(\varphi(t))\cos(2\pi f_н t) + a(t)\sin(\varphi(t))\sin(2\pi f_н t) \text{ или}$$

$$y(t) = i(t)\cos(2\pi f_н t) + q(t)\sin(2\pi f_н t),$$

где $i(t) = a(t)\cos(\varphi(t))$, а $q(t) = a(t)\sin(\varphi(t))$.

Функция $i(t)$ называется *синфазной* составляющей (так как она находится в фазе с исходным сигналом), а функция $q(t)$ — *квадратурной* составляющей (она отличается на 90 градусов от исходного сигнала, так как $\sin(x) = \cos(90° + x)$ модулированного сигнала).

Случай частотной модуляции, когда известна зависимость $f_m(t)$, можно свести к фазовой модуляции. По определению мгновенная частота равна производной мгновенной фазы по времени:

$$f(t) = 1/2\pi\, d\varphi(t)/dt.$$

Соответственно, закон изменения мгновенной фазы определяется по известному закону изменения мгновенной частоты путем интегрирования:

$$\varphi(t) = \varphi 0 + 2\pi \int_0^t f_m(\tau)\, d\tau.$$

Подставляя это выражение мгновенной фазы в исходную формулу для модулированного сигнала, мы сводим случай частотного модулирования к случаю фазового модулирования.

Представление модулированного сигнала в виде суммы произведения синфазной и квадратурной составляющих на синусоидальные сигналы несущей частоты дают идею реализа-

ции **квадратурного модулятора** в виде набора блоков, таких как смесители и сумматоры. Типовая схема квадратичного модулятора показана на рис. 7.10.

Первый блок отображает значение очередной порции из четырех бит входной последовательности двоичного кода (b1, b2, b3, b4) в пару значений амплитуды и фазы модулирующего сигнала $\{a_i, \varphi_j\}$, то есть отображает значение входной четверки бит из диапазона 0–15 на соответствующий символ кода 16-QAM. Этот же блок генерирует два синусоидальных сигнала, синфазный $i(t) = a_i(t)\cos(\varphi_j(t))$ и квадратурный $q(t) = a_i(t)\sin(\varphi_j(t))$, используя найденные амплитуду a_i и фазу φ_j.

Синфазный и квадратурный сигналы подаются на смесители, которые перемножают эти сигналы на сигнал несущей частоты, вырабатываемой генератором Г. На смеситель, перемножающий квадратурный сигнал, сигнал несущей частоты подается через фазовращатель, изменяющей фазу на 90 градусов для получения синусоиды из сигнала косинусоиды.

Затем оба сигнала произведений подаются на входы сумматора, на выходе которого получается искомый модулированный сигнал.

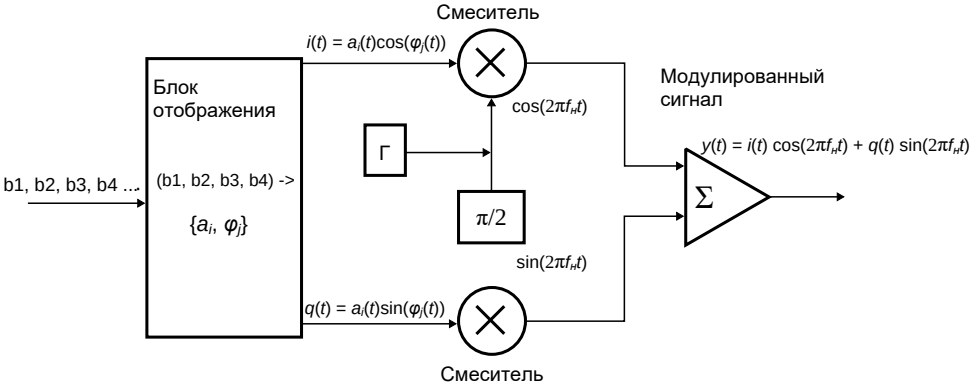


Рис. 7.10. Схема квадратурного модулятора

Реализация квадратурного модулятора может отличаться в зависимости от того, каким способом выполняет свои функции каждый блок — аналоговым или цифровым.

На практике применяются три варианта реализации квадратичного модулятора:

- **Аналоговый.** Все блоки модулятора являются аналоговыми, только блок отображения имеет и цифровую часть (выделяющую четверки бит и находящую соответствующие значения амплитуды и фазы) и аналоговую (генерирующую синфазные и квадратурные сигналы)
- **Цифровой.** Все блоки работают как цифровые процессоры сигналов, только выходной модулированный сигнал преобразуется в аналоговый с помощью быстрого обратного преобразования Фурье. В этом случае дискретные значения синусоид либо вычисляются «на лету», либо полная предварительно вычисленная кривая каждого символа хранится в памяти модулятора (обычно в виде набора частот) и вызывается из нее для преобразования в аналоговую форму. Последний способ получил название прямого цифрового синтеза (Direct Digital Syntheses, DDS).

- **Гибридный.** Синфазная и квадратурная составляющие вычисляются в цифровой форме, затем преобразуются в аналоговую и перемножаются на несущую частоту уже в аналоговой форме.

Для **демодуляции** сигнала также применяется квадратурное преобразование — умножение полученного сигнала на сигнал несущей частоты для выделения синфазной составляющей и на сдвинутый на 90 градусов сигнал несущей частоты для выделения квадратурной составляющей. Так как при умножении в результирующем сигнале образуются высокочастотные составляющие удвоенной частоты несущей и полезный низкочастотный сигнал, то результат умножения пропускают через низкочастотный фильтр, после которого и получаются синфазная и квадратурная составляющие сигнала.

Затем эти две составляющие преобразуют в значение модулирующего параметра — амплитуды, частоты или фазы сигнала либо их комбинации. Например, при фазовой модуляции значение фазы определяется из тригонометрического соотношения:

$$\varphi = \arctan(q/i).$$

Обнаружение и коррекция ошибок

Методы обнаружения ошибок

Методы обнаружения ошибок основаны на передаче в составе блока данных избыточной служебной информации, по которой можно судить с некоторой степенью вероятности о достоверности принятых данных. В сетях с коммутацией пакетов такой единицей информации может быть протокольная единица данных любого уровня, но для определенности будем считать, что мы контролируем кадры.

Избыточную служебную информацию принято называть **контрольной суммой** или **контрольной последовательностью кадра** (Frame Check Sequence, FCS). Контрольная сумма вычисляется как функция от основной информации, причем *не обязательно путем суммирования*. Принимающая сторона повторно вычисляет контрольную сумму кадра по известному алгоритму и в случае ее совпадения с контрольной суммой, вычисленной передающей стороной, делает вывод о том, что данные были переданы через сеть корректно. Рассмотрим несколько распространенных алгоритмов вычисления контрольной суммы, отличающихся вычислительной сложностью и способностью обнаруживать ошибки в данных.

Контроль по паритету представляет собой наиболее простой метод контроля данных. В то же время это наименее мощный алгоритм контроля, позволяющий обнаруживать только одиночные ошибки в проверяемых данных. Метод заключается в суммировании по модулю 2 всех битов контролируемой информации. Нетрудно заметить, что для информации, состоящей из нечетного числа единиц, контрольная сумма всегда равна 1, а при четном числе единиц — 0. Например, для данных 100101011 результатом контрольного суммирования будет значение 1. Результат суммирования также представляет собой один дополнительный бит данных, который пересылается вместе с контролируемой информацией. При искажении в процессе пересылки любого одного бита исходных данных (или контрольного разряда) результат суммирования будет отличаться от принятого контрольного разряда, что говорит об ошибке. Однако двойная ошибка, например 110101010, будет неверно принята за корректные данные. Поэтому контроль по паритету применяется к небольшим

порциям данных, как правило, к каждому байту, что дает для этого метода коэффициент избыточности 1/8. Метод редко используется в компьютерных сетях из-за значительной избыточности и невысоких диагностических возможностей.

Вертикальный и горизонтальный контроль по паритету представляет собой модификацию описанного метода. Его отличие состоит в том, что исходные данные рассматриваются в виде матрицы, строки которой составляют байты данных. Контрольный разряд подсчитывается отдельно для каждой строки и для каждого столбца матрицы. Этот метод позволяет обнаруживать бóльшую часть двойных ошибок, однако он обладает еще большей избыточностью. На практике этот метод сейчас также почти не применяется при передаче информации по сети.

Циклический избыточный контроль (Cyclic Redundancy Check, CRC) является в настоящее время наиболее популярным методом контроля в вычислительных сетях (и не только в сетях — например, этот метод широко применяется при записи данных на гибкие и жесткие диски). Метод основан на представлении исходных данных в виде одного многоразрядного двоичного числа. Например, кадр стандарта Ethernet, состоящий из 1024 байт, рассматривается как одно число из 8192 бит. Контрольной информацией считается остаток от деления этого числа на известный делитель *R*. Обычно в качестве делителя выбирается семнадцати- или тридцатитрехразрядное число, чтобы остаток от деления имел длину 16 разрядов (2 байта) или 32 разряда (4 байта). При получении кадра данных снова вычисляется остаток от деления на тот же делитель *R*, но при этом к данным кадра добавляется содержащаяся в нем контрольная сумма. Если остаток от деления на *R* равен нулю, то делается вывод об отсутствии ошибок в полученном кадре, в противном случае кадр считается искаженным.

Этот метод обладает более высокой вычислительной сложностью, но его диагностические возможности гораздо выше, чем у методов контроля по паритету. Метод CRC позволяет обнаруживать все одиночные ошибки, двойные ошибки и ошибки в нечетном числе битов. Кроме того, метод обладает невысокой степенью избыточности. Например, для кадра Ethernet размером 1024 байта контрольная информация длиной 4 байта составляет только 0,4 %.

Методы коррекции ошибок

Техника кодирования, которая позволяет приемнику не только выявить в принятых данных ошибки, но и исправить их, называется **прямой коррекцией ошибок** (Forward Error Correction, FEC). Коды, которые обеспечивают прямую коррекцию ошибок, требуют введения большей избыточности в передаваемые данные, чем коды, только обнаруживающие ошибки. При применении любого избыточного кода не все комбинации кодов являются разрешенными. Например, контроль по паритету делает разрешенными только половину кодов. Если мы контролируем три информационных бита, то разрешенными 4-битными кодами с дополнением до нечетного количества единиц будут следующие:

000 1, 001 0, 010 0, 011 1, 100 0, 101 1, 110 1, 111 0,

то есть 8 кодов из 16 возможных.

Чтобы оценить количество дополнительных битов, требуемых для исправления ошибок, нужно знать так называемое расстояние Хемминга между разрешенными комбинациями

кода. **Расстоянием Хемминга** называется минимальное число битовых разрядов, в которых отличается любая пара разрешенных кодов. Для схем контроля по паритету расстояние Хемминга равно 2.

Если представить, что мы сконструировали избыточный код с расстоянием Хемминга, равным n, то такой код будет в состоянии распознавать $(n-1)$-кратные ошибки и исправлять $(n-1)/2$-кратные ошибки. Так как коды с контролем по паритету имеют расстояние Хемминга, равное 2, то они могут только обнаруживать однократные ошибки и не могут исправлять ошибки.

Коды Хемминга эффективно обнаруживают и исправляют изолированные ошибки, то есть отдельные искаженные биты, которые разделены большим количеством корректных битов. Однако при появлении длинной последовательности искаженных битов (пульсации ошибок) коды Хемминга не работают.

Пульсации ошибок характерны для *беспроводных каналов*, в которых применяют **сверточные коды**. Поскольку для распознавания наиболее вероятного корректного кода в этом методе применяется решетчатая диаграмма, то такие коды также называют **решетчатыми**. Эти коды используются не только в беспроводных каналах, но и в модемах.

Методы прямой коррекции ошибок особенно эффективны для технологий физического уровня, которые не поддерживают сложные процедуры повторной передачи данных в случае их искажения. Примерами таких технологий являются SDH и OTN, рассматриваемые в главе 8 этой книги.

Кодирование аналоговой информации

Кодирование аналоговой информации аналоговыми сигналами

Исторически модуляция начала применяться именно для кодирования *аналоговой информации* и только потом — для дискретной.

Необходимость в модуляции аналоговой информации возникает, когда нужно передать *низкочастотный* аналоговый сигнал через канал, находящийся в высокочастотной области спектра. Пример — передача голоса по радио или телевидению. Голос имеет спектр шириной примерно в 10 кГц, а радиодиапазоны включают гораздо более высокие частоты, от 30 кГц до 300 мГц. Еще более высокие частоты используются в телевидении. Очевидно, что непосредственно голос через такую среду передать нельзя.

Для решения проблемы амплитуду высокочастотного несущего сигнала изменяют (модулируют) в соответствии с изменением низкочастотного голосового сигнала (рис. 7.11). При этом спектр результирующего сигнала попадает в нужный высокочастотный диапазон. Такой тип модуляции называется **амплитудной модуляцией** (Amplitude Modulation, AM).

В качестве информационного параметра используют не только амплитуду несущего синусоидального сигнала, но и частоту. В подобных случаях мы имеем дело с **частотной модуляцией** (Frequency Modulation, FM). Заметим, что при модуляции аналоговой информации фаза как информационный параметр не применяется.

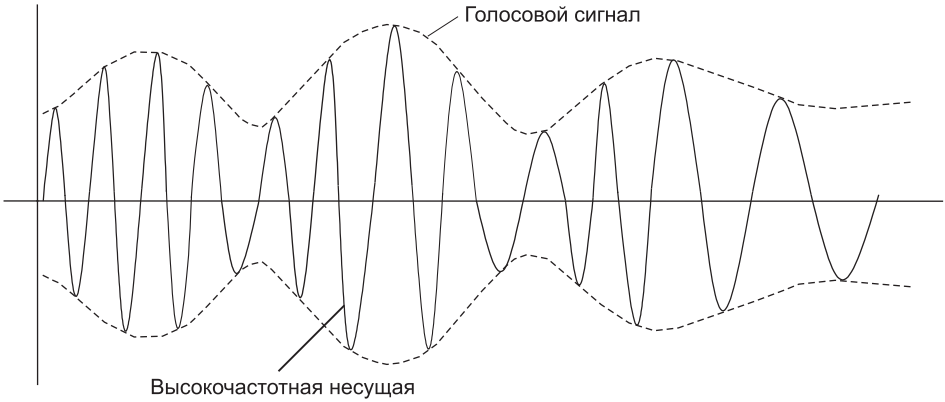


Рис. 7.11. Модуляция голосовым сигналом

Кодирование аналоговой информации дискретными сигналами

В предыдущем разделе мы познакомились с преобразованием дискретной формы представления информации в аналоговую. В этом разделе рассматривается решение обратной задачи — передачи аналоговой информации в дискретной форме. Такая задача решается в системах цифровой телефонии (радио, телевидения).

Начиная с 60-х годов прошлого века голос начал передаваться по телефонным сетям в цифровой форме, то есть в виде последовательности единиц и нулей. Основной причиной такого перехода является невозможность улучшения качества данных, переданных в аналоговой форме, если они существенно исказились при передаче. Сам аналоговый сигнал не дает указаний ни на то, что произошло искажение, ни на то, как его исправить, поскольку форма сигнала может быть любой, в том числе той, которую зафиксировал приемник. Улучшение качества аналоговых телефонных линий, особенно территориальных, требует огромных усилий и капиталовложений. Поэтому на смену аналоговой технике записи и передачи звука и изображений пришла техника цифровая.

В этой технике используется так называемая **дискретная модуляция** исходных непрерывных во времени аналоговых процессов. Амплитуда исходной непрерывной функции измеряется с заданным периодом, что обеспечивает *дискретизацию по времени*. Затем каждый замер представляется в виде двоичного числа определенной разрядности, что означает *дискретизацию по значениям* — непрерывное множество возможных значений амплитуды заменяется дискретным множеством ее значений.

Устройство, которое выполняет подобную функцию, называется **аналого-цифровым преобразователем** (АЦП). Затем замеры передаются по линиям связи в виде последовательности единиц и нулей. При этом применяются те же методы кодирования (с ними мы познакомимся позднее), что и при передаче изначально дискретной информации. На приемной стороне линии коды преобразуются в исходную последовательность битов, а специальная аппаратура, называемая **цифро-аналоговым преобразователем** (ЦАП), производит демодуляцию оцифрованных амплитуд, восстанавливая исходную непрерывную

функцию времени. Устройство, которое может выполнять функции как АЦП, так и ЦАП, называется **кодеком**.

Дискретная модуляция основана на теореме отображения *Котельникова — Найквиста* , в соответствии с которой аналоговая непрерывная функция, переданная в виде последовательности ее дискретных по времени значений, может быть точно восстановлена, если частота дискретизации Fд была в два или более раз выше, чем частота самой высокой гармоники спектра исходной функции Fmax:

$$Fд \geq 2Fmax.$$

Если это условие не соблюдается, то восстановленная функция будет отличаться от исходной. Существуют исключения из этой теоремы, они соответствуют случаям, когда известна дополнительная информация о спектре исходной функции. Так, если известно, что спектр частот аналогового сигнала заключен в границы некоторой полосы частот B = Fmax — Fmin, то для того, чтобы аналоговый сигнал мог быть восстановлен, частота дискретизации должна превышать удвоенное значение ширины полосы частот B, то есть **Fд** ≥ 2 B.

Преимуществом цифровых методов записи, воспроизведения и передачи аналоговой информации является возможность контроля достоверности считанных с носителя или полученных по линии связи данных. Для этого можно применять те же методы, что и в случае компьютерных данных — вычисление контрольной суммы, повторная передача искаженных кадров, применение самокорректирующихся кодов.

Для представления голоса в цифровой форме используются различные методы его дискретизации. Наиболее простой метод оцифровывания голоса, в котором применяется частота квантования амплитуды звуковых колебаний в 8000 Гц, уже был кратко рассмотрен в главе 3 (раздел «Коммутация каналов») и известен как метод **импульсно-кодовой модуляции** (Pulse Code Modulation, **PCM**).

Обоснование выбранной частоты квантования в методе PCM достаточно простое. В аналоговой телефонии для передачи голоса используется диапазон от 300 до 3400 Гц, который достаточно качественно передает все основные гармоники собеседников. В соответствии с теоремой Найквиста — Котельникова для качественной передачи голоса достаточно выбрать частоту дискретизации, в два раза превышающую самую высокую гармонику непрерывного сигнала, то есть 2 × 3400 = 6800 Гц. Выбранная в действительности частота дискретизации 8000 Гц обеспечивает некоторый запас качества. В методе PCM обычно используется 7 или 8 бит кода для представления амплитуды одного замера. Соответственно, это дает 127 или 256 градаций звукового сигнала, что оказывается вполне достаточно для качественной передачи голоса.

При использовании метода PCM для передачи одного голосового канала необходима пропускная способность 56 или 64 Кбит/с в зависимости от того, каким количеством битов представляется каждый замер. Если для этих целей применяется 7 бит, то при частоте передачи замеров в 8000 Гц получаем: 8000 × 7 = 56 000 бит/с, или 56 Кбит/с; для случая 8 бит: 8000 × 8 = 64000 бит/с, или 64 Кбит/с.

Стандартным является цифровой канал 64 Кбит/с (известен также как **элементарный канал цифровых телефонных сетей**); канал 56 Кбит/с применялся на ранних этапах существования цифровой телефонии, когда один бит из байта, отведенного для пере-

дачи данных, изымался для передачи номера вызываемого абонента. Передача непрерывного сигнала в дискретном виде требует от сетей жесткого соблюдения временного интервала в **125 мкс** (соответствующего частоте дискретизации 8000 Гц) между соседними замерами, то есть требует синхронной передачи данных между узлами сети. При отсутствии синхронности прибывающих замеров исходный сигнал восстанавливается неверно, что приводит к искажению голоса, изображения или другой мультимедийной информации, передаваемой по цифровым сетям. Так, искажение синхронизации в 10 мс может привести к эффекту эха, а сдвиги между замерами в 200 мс приводят к невозможности распознавания произносимых слов. Потеря одного замера при соблюдении синхронности между остальными замерами практически не сказывается на воспроизводимом звуке. Это достигается за счет сглаживающих устройств в цифро-аналоговых преобразователях, работа которых основана на свойстве инерционности любого физического сигнала — амплитуда звуковых колебаний не может мгновенно измениться на значительную величину.

Мультиплексирование и коммутация

Для повышения эффективности использования физической линии связи применяют различные методы разделения этой линии между несколькими логическими каналами (потоками) данных пользователей сети. Действительно, услуги оптоволоконной линии связи, по которой параллельно передаются данные сеансов связи многих пар пользователей, будут гораздо доступнее, чем при ее индивидуальном использовании. При этом требуют решения следующие задачи:

- *мультиплексирование*, то есть образование из нескольких потоков общего агрегированного потока;
- *демультиплексирование* — разделение агрегированного потока на составляющие его потоки;
- *коммутация* — переключение потоков между портами сетевых устройств для соединения пользователей сети.

Очевидно, что способы реализации мультиплексирования, демультиплексирования и коммутации в одной и той же сети должны быть согласованными и построенными на единых принципах. Такой общей основой для каждой «тройки» задач может служить один из следующих методов мультиплексирования:

- частотное мультиплексирование (Frequency Division Multiplexing, FDM);
- волновое мультиплексирование (Wave Division Multiplexing, WDM);
- временно́е мультиплексирование (Time Division Multiplexing, TDM).

Методы FDM и WDM пригодны исключительно для сетей с *коммутацией каналов*. Временно́е мультиплексирование TDM имеет две существенно отличающиеся разновидности — асинхронное и синхронное разделение времени. Асинхронный метод TDM является основой *пакетных* сетей, а синхронный вариант TDM используется в сетях с *коммутацией каналов*.

Применение того или иного метода мультиплексирования влечет за собой применение метода коммутации, базирующегося на том же принципе.

Мультиплексирование и коммутация на основе методов FDM и WDM

Техника **частотного мультиплексирования (FDM)** была разработана для телефонных сетей, но она применяется и для других видов сетей, например беспроводных. Основная идея метода состоит в выделении каждому соединению собственного диапазона (полосы) частот в общей полосе пропускания линии связи. На основе диапазона создается **канал**. Данные, передаваемые в канале, модулируются с помощью одного из описанных ранее методов с использованием несущей частоты, принадлежащей диапазону канала. Мультиплексирование выполняется с помощью смесителя частот, демультиплексирование — с помощью узкополосного фильтра, по ширине равного ширине диапазона канала. Рассмотрим особенности этого вида мультиплексирования на примере *телефонной сети*.

На входы FDM-мультиплексора поступают исходные сигналы от абонентов телефонной сети (в нашем примере их 12). Мультиплексор переносит сигнал каждого канала в выделенную каналу полосу частот за счет модуляции новой несущей частоты, принадлежащей этой полосе. Чтобы низкочастотные составляющие сигналов разных каналов не смешивались между собой, полосы делают шириной в 4 кГц, а не в 3,1 кГц (ширина полосы пропускания канала тональной частоты), оставляя между ними страховочный промежуток в 900 Гц (рис. 7.12).

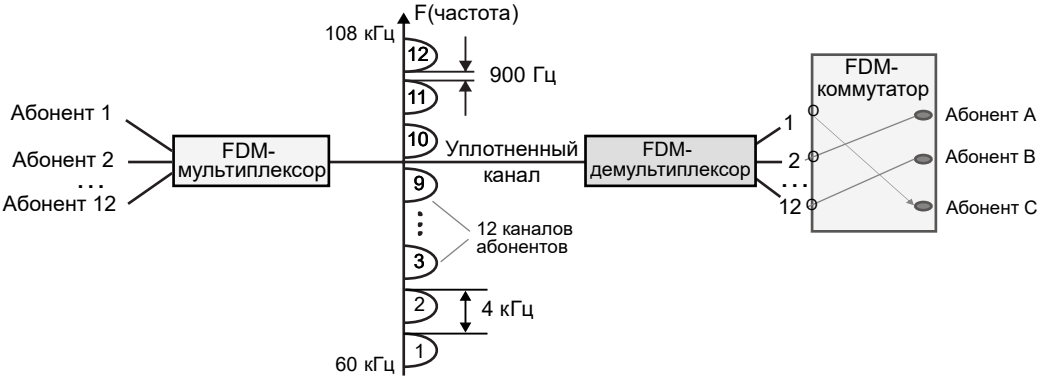


Рис. 7.11. Мультиплексирование/демультиплексирование и коммутация на основе FDM

В линии связи между двумя FDM-устройствами одновременно передаются сигналы всех 12 абонентских каналов, но каждый из них занимает *свою* полосу частот. Такой канал называют **уплотненным**.

Выходной FDM-демультиплексор выделяет модулированные сигналы каждой несущей частоты и передает их на коммутирующий блок, который переключает их на соответствующие выходные каналы, к которым непосредственно подключены абонентские телефоны.

FDM-коммутаторы могут выполнять как динамическую, так и постоянную коммутацию. При *динамической коммутации* один абонент инициирует соединение с другим абонентом, посылая в сеть его номер, а коммутатор выделяет данному абоненту одну из свободных полос своего уплотненного канала *на время сеанса связи*. Типичным представителем динами-

ческого коммутатора является коммутатор аналоговой телефонной станции. При *постоянной коммутации* администратор сети закрепляет полосу за абонентом на *длительный срок*.

Принцип коммутации на основе разделения частот остается неизменным и в других, отличных от телефонных, сетях. Меняются только границы полос, выделяемых отдельному абонентскому каналу, а также количество низкоскоростных каналов в высокоскоростном канале.

В методе **волнового мультиплексирования** (технология **WDM** с ее двумя разновидностями **CWDM** и **DWDM**) используется тот же принцип частотного разделения каналов, но только в другой области электромагнитного спектра. Для организации WDM-каналов в волоконно-оптическом кабеле задействуют волны инфракрасного диапазона длиной от 850 до 1565 нм. В магистральном канале мультиплексируется до нескольких десятков спектральных каналов. Отличие сетей WDM от сетей FDM заключается в предельных скоростях передачи информации. Если сети FDM обычно обеспечивают на магистральных каналах одновременную передачу до 600 разговоров, что соответствует суммарной скорости в 36 Мбит/с, то сети DWDM обладают пропускной способностью до сотен гигабитов и даже нескольких терабитов в секунду (подробнее о технологиях DWDM и CWDM см. главу 9 и разделы главы 10, посвященные высокоскоростным версиям Ethernet).

Мультиплексирование и коммутация на основе метода TDM

FDM-мультиплексирование разрабатывалось в расчете на передачу голосовых *аналоговых* сигналов. Переход к *цифровой* форме представления голоса стимулировал разработку новой техники мультиплексирования, ориентированной на дискретный характер передаваемых данных и носящей название **мультиплексирования с разделением времени** или **временно́го мультиплексирования (TDM)**.

Принцип временно́го мультиплексирования заключается в выделении канала каждому соединению на определенный период времени. Применяются два типа временно́го мультиплексирования — асинхронный и синхронный. С **асинхронным режимом TDM** мы уже знакомы — он применяется в сетях с коммутацией пакетов. Каждый пакет занимает канал определенное время, необходимое для его передачи между конечными точками канала. Между различными информационными потоками нет синхронизации, и каждый пользователь пытается занять канал тогда, когда у него возникает потребность в передаче информации. Если такой возможности нет, то пакет целиком *буферизуется* и ожидает в очереди буфера момента освобождения канала. Таким образом, данные пользователей сети с коммутацией пакетов разделяют канал во времени *асинхронно*, не координируя свои действия друг с другом.

Рассмотрим теперь **синхронный режим TDM**[1]. В этом режиме доступ всех пользовательских информационных потоков к разделяемому каналу *синхронизируется* таким образом, чтобы каждый информационный поток периодически получал канал в свое распоряжение

[1] Если аббревиатура TDM используется без уточнения режима работы, то она всегда обозначает синхронный режим TDM.

на фиксированный и одинаковый для всех потоков промежуток времени, называемый **тайм-слотом**.

Аппаратура разделяемого канала последовательно предоставляет тайм-слот очередному пользовательскому потоку, который в течение этого кванта времени передает в разделяемый канал порцию данных (например, один байт). Период времени, в течение которого все пользовательские потоки получат по одному тайм-слоту доступа к каналу, называется **циклом** работы канала и аппаратуры, его образующей. Таким образом, тайм-слот равен величине цикла, деленной на количество мультиплексируемых пользовательских потоков. Например, при мультиплексировании оцифрованных голосовых каналов цикл равен 125 мкс и при 24 абонентских каналах, обслуживаемых мультиплексором, тайм-слот составит 5,2 мкс. Данные, принятые от всех пользовательских потоков за один цикл, образуют **кадр**. Он может включать, кроме информации пользовательских потоков, и служебную информацию, например, синхробиты или синхробайты, позволяющие устройствам TDM распознавать начало каждого кадра и, следовательно, правильно соотносить данные каждого тайм-слота с абонентом сети. Байт, считываемый из определенного пользовательского потока в каждом цикле, занимает в кадре *всегда одну и ту же позицию*. Тем самым порядковый номер байта в кадре (или однозначно связанные с ним порядковый номер тайм-слота или номер интерфейса) является *адресом* абонентского потока в канале. Именно на этих адресах основана коммутация в сетях TDM. Принцип работы мультиплексора TDM представлен на рис. 7.13.

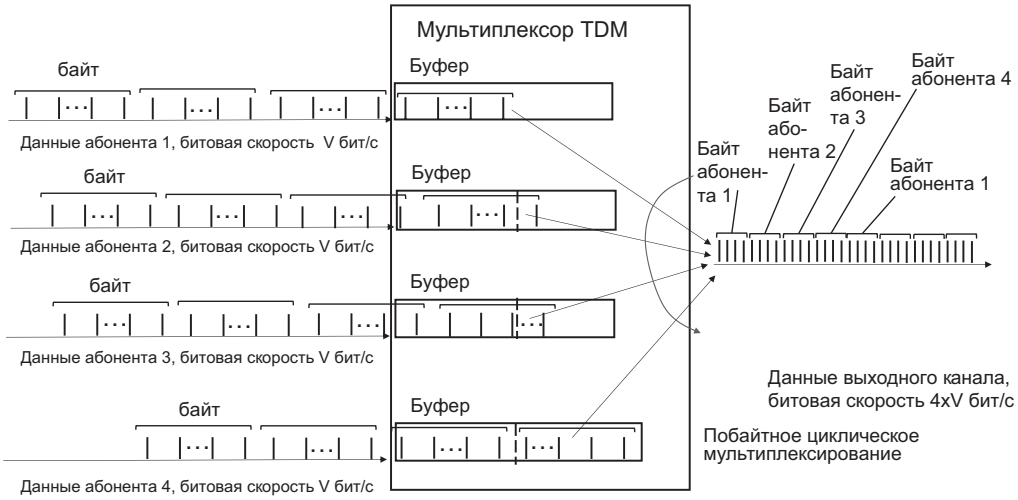


Рис. 7.13. Мультиплексирование голосовых потоков на основе техники TDM

Мультиплексор входит в состав комбинированного терминального устройства, включающего также *кодек*, который оцифровывает голосовые сигналы четырех абонентов, поступающие в аналоговом виде на его входы. Кодек производит по одному байту цифрового кода голоса каждого абонента каждые 125 мкс, так что на каждый вход мультиплексора поступает битовый поток со скоростью 64 Кбит/с. Мультиплексор помещает поочередно по одному байту данных от каждого из четырех пользователей на выходной канал, работающий со скоростью 4 × 64 = 256 Кбит/с. Величина тайм-слота, выделяемого каждому абоненту в течение цикла 125 мкс, равна 125/4 = 31,25 мкс.

Кодек и мультиплексор работают согласованно и *синхронно*, так как являются блоками одного и того же комбинированного устройства. К моменту завершения первого тайм-слота кодек производит байт данных второго абонента и помещает его в буфер мультиплексора, отведенный для данных второго абонента. Мультиплексор извлекает его из буфера и передает его побитно на выходной канал в течение второго тайм-слота цикла. Этот процесс повторяется с байтами третьего и четвертого абонентов, на чем цикл завершается. За время цикла мультиплексор передает на выходной канал байты всех абонентов, тем самым завершив передачу кадра выходного канала.

Демультиплексирование кадра агрегированного канала выполняется в обратном порядке. За один цикл работы демультиплексора байты по очереди принимаются из высокоскоростного канала и помещаются в буферы абонентских каналов. Биты первого байта кадра принимаются в течение первого тайм-слота, второго — в течение второго тайм-слота и т. д. После принятия очередного байта кодек производит цифро-аналоговое преобразование кода и передает звуковой аналоговый сигнал соответствующему абоненту. Циклы повторяются и сигналы каждому абоненту поступают с частотой 8 КГц, достаточной, как мы знаем, для качественного воспроизведения голоса.

Теперь рассмотрим, каким образом происходит *коммутация* каналов в сети, работающей по принципу синхронного TDM.

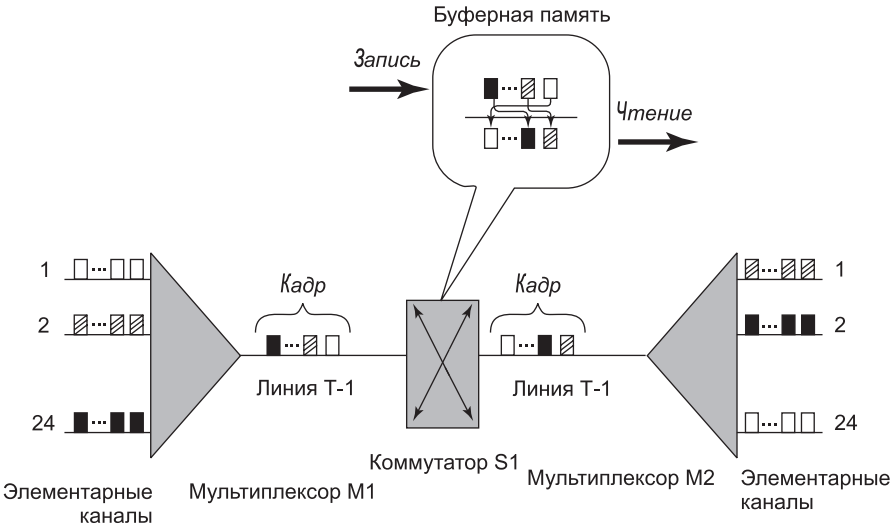


Рис. 7.14. Коммутация на основе разделения канала во времени

На рис. 7.14 показан фрагмент цифровой телефонной сети, использующей технику TDM. Этот фрагмент состоит из коммутатора S1 и мультиплексоров M1 и M2, которые мультиплексируют и демультиплексируют оцифрованные голосовые данные 24 абонентов в соответствии с техникой TDM.

Абоненты подключены к портам мультиплексоров, и номера портов подключения являются их идентификаторами, на основании которых в сети выполняется коммутация абонентов. Байты, принятые от всех 24 абонентов в течение одного цикла, упаковываются

в кадр в порядке, соответствующем порядку абонентских портов. Кадр снабжен коротким заголовком (на рисунке не показан), позволяющим распознавать начало кадра, что дает возможность определить, какой байт кадра какому тайм-слоту и соответственно какому абоненту принадлежит.

Задача коммутатора S1 состоит в следующем. Абонент, подключенный к абонентскому каналу 1 мультиплексора M1, должен быть связан с абонентом, подключенным к абонентскому каналу 24 мультиплексора M2, абонент 2 мультиплексора M1 — с абонентом 1 мультиплексора M2, абонент 24 мультиплексора M1 — с абонентом 2 мультиплексора M2.

Коммутатор S1 принимает кадр по скоростному каналу от мультиплексора M1 и записывает каждый байт из него в отдельную ячейку своей буферной памяти именно в том порядке, в котором байты упакованы в кадр. Для выполнения коммутации байты извлекаются из буферной памяти не в порядке поступления, а в том порядке, который соответствует поддерживаемым в сети соединениям абонентов. В рассматриваемом примере первым из буферной памяти должен быть извлечен байт 2, вторым — байт 24, а последним — байт 1. «Перемешивая» нужным образом байты в кадре, коммутатор обеспечивает требуемое соединение абонентов в сети. Мультиплексор M2 решает обратную задачу — он разбирает кадр на байты и распределяет их по своим выходным каналам, при этом он также считает, что порядковый номер байта в кадре соответствует номеру выходного канала.

Мы рассмотрели операции мультиплексирования, коммутации и демультиплексирования в одном направлении — от абонентов мультиплексора M1 к абонентам мультиплексора M2. В обратном направлении все операции выглядят аналогично.

Работа TDM-оборудования напоминает работу сетей с коммутацией пакетов, так как каждый байт данных можно считать некоторым элементарным пакетом. Однако в отличие от пакета компьютерной сети «пакет» сети TDM не имеет индивидуального адреса. Его адресом является порядковый номер в кадре или номер выделенного тайм-слота в мультиплексоре или коммутаторе. Сети, использующие технику TDM, требуют синхронной работы всего оборудования, что и определило второе название этой техники — **синхронный режим передачи** (Synchronous Transfer Mode, STM).

Нарушение синхронности разрушает требуемую коммутацию абонентов, так как при этом изменяется относительное положение тайм-слота, а значит, теряется адресная информация. Поэтому оперативное перераспределение тайм-слотов между различными каналами в TDM-оборудовании невозможно. Если в каком-то цикле работы мультиплексора тайм-слот одного из каналов оказывается избыточным, поскольку на входе этого канала в данный момент нет данных для передачи (например, абонент телефонной сети молчит), то он передается пустым.

ГЛАВА 8 Технологии первичных сетей OTN

Особенности первичных сетей

Первичные сети предназначены для создания коммутируемой инфраструктуры, с помощью которой можно достаточно быстро и гибко организовать **постоянный** канал обмена данными между двумя пользовательскими устройствами, подключенными к такой сети.

Основные свойства первичных сетей состоят в следующем:

- Они основаны на технике *коммутации каналов*.
- Каналы первичных сетей являются *статическими* (постоянными), в отличие от динамических каналов компьютерных и телефонных сетей, предоставляя услугу канала «*точка-точка*» между двумя определенными абонентами на протяжении большого промежутка времени — месяц, год.
- На основе каналов, образованных первичными сетями, работают *вторичные наложенные* (overlay) компьютерные или телефонные сети.
- Каналы первичных сетей являются магистральными каналами, они образуют высокоскоростные каналы — *магистрали* — между крупными центрами, в которых сосредоточено большое количество потребителей, которым нужны соединения между этими центрами. Магистральные каналы также называют агрегатными каналами, так как они агрегируют большое количество пользовательских потоков данных.

Каналы, предоставляемые первичными сетями своим пользователям, отличаются высокой пропускной способностью — до 100–800 Гбит/с, большой протяженностью, составляющей сотни и тысячи километров, а также высоким качеством, что выражается в очень низком проценте битовых ошибок. Для обеспечения таких скоростей и таких расстояний в наибольшей степени подходит оптоволоконный кабель, поэтому название **«оптические сети»** прочно закрепилось в качестве еще одного названия первичных сетей, несмотря на то что оптоволоконные кабели применяются и в других типах сетей, например в сетях Ethernet.

Первичные сети также называют **транспортными сетями**, так как они предоставляют своим клиентам только транспортные услуги, передавая пользовательскую информацию «из пункта А в пункт Б» без ее изменения. Пользовательским оборудованием, использующим каналы первичных сетей, являются как телефонные станции (для которых эти сети и были первоначально придуманы), так и маршрутизаторы компьютерных сетей. Чтобы использовать канал некоторой первичной сети для соединения двух маршрутизаторов, последние должны иметь физические порты, соответствующие стандарту данной первичной сети.

Организация первичных сетей существенно отличается от организации компьютерных сетей, работающих на основе принципа коммутации пакетов. Нужно предупредить читателя,

что очень часто его знания из области сетей с коммутацией пакетов не будут помогать ему понять механизмы работы первичной сети — настолько они различны.

В частности, семиуровневая модель OSI, хорошо помогающая понять, как протоколы различных уровней обслуживают друг друга, чтобы доставить очередную порцию данных пользователю, здесь неприменима. С ее точки зрения, все механизмы и процедуры первичной сети относятся к одному уровню — физическому. Даже кадр в технологиях первичных сетей имеет мало общего с кадром пакетных сетей.

Существует два типа первичных сетей:

- На основе синхронного временного мультиплексирования TDM, рассмотренного нами в последнем разделе предыдущей главы. К этому типу технологий относятся технологии PDH, SDH и OTN.
- На основе волнового мультиплексирования DWDM, которое создает основу для работы технологии синхронного TDM, предоставляя ей набор волновых каналов в качестве передающей среды.

В этой главе мы изучим современную технологию первого типа — OTN, а технологии DWDM посвящена следующая глава.

Технологии PDH и SDH мы рассмотрим кратко, так как они практически уже не применяются в новых проектах и представляют только исторический интерес, показывая эволюцию этого типа сетей.

Топология и типы оборудования сетей синхронного TDM

Рассмотрим сначала общие свойства сетей, использующих технику синхронного временно́го мультиплексирования TDM.

Первичные сети, работающие на основе синхронного TDM, являются системами, *синхронно* передающими байт за байтом и кадр за кадром пользовательских данных от входного интерфейса первого мультиплексора сети до выходного интерфейса последнего мультиплексора на пути от одного пользователя до другого.

Первичные сети TDM работают *циклично:* байты и кадры одного и того же соединения появляются на выходе мультиплексора в одном и том же тайм-слоте цикла, позиция тайм-слота в периоде цикла является неявным адресом соединения. В отличие от одноуровневой схемы мультиплексирования, изображенной на рис. 7.12, мультиплексирование в сетях SDH и OTN иерархическое: кадры одного уровня упаковываются в кадры более высокого уровня и таких уровней несколько. Соответственно, скорости передачи данных на каждом уровне отличаются. С самой высокой скоростью передаются данные самого верхнего уровня иерархии инкапсуляции кадров, поэтому, описывая технологию SDH или OTN, говорят об их иерархии скоростей.

В сети синхронного TDM не бывает пауз между пользовательскими данными, как это происходит в компьютерной сети, когда компьютер пользователя может какое-то время не вырабатывать данные для передачи по сети. Слаженную работу всех устройств первичной сети можно сравнить с работой конвейера: в каждом такте своей работы конвейер перемещает объект, над которым производятся операции сборки, от одного рабочего места к другому. Если рабочий не успевает выполнить свою операцию за время такта конвейера,

то объект уходит от него к следующему рабочему месту без какой-то нужной детали, и исправить это уже невозможно. Также и в первичной сети: если очередной байт пользовательских данных не успел поступить на входной интерфейс мультиплексора сети вовремя из-за рассинхронизации оборудования пользователя и мультиплексора, то в этом такте работы мультиплексора очередной байт на выходном интерфейсе появляется с «пустым» содержанием. Таким образом, он не несет пользовательской информации, а появляется на выходном интерфейсе только потому, что мультиплексор должен передать байт в этом такте, поскольку это предусмотрено принципом синхронной работы сети.

Наиболее простой топологией первичной сети TDM является совокупность нескольких отдельных линий связи «точка-точка», каждая из которых связывает две географически разнесенные группы пользователей (рис. 8.1, *а*). Основными узлами такой сети являются мультиплексоры и демультиплексоры, функции которых часто совмещаются в одном устройстве, называемом также **терминальным мультиплексором** (Terminal Multiplexer, **TM**), так как они завершают (терминируют) линию связи. Терминальный мультиплексор объединяет потоки данных на входах своих клиентских интерфейсов в один более скоростной поток, а перед передачей их на выходные интерфейсы разбирает их на отдельные потоки пользователей.

В некоторых случаях возникает необходимость подключения пользователей, расположенных не в конечных узлах линии связи, «точка-точка», а в некотором *промежуточном* пункте. Например, в пункте *R* группа пользователей имеет потребность в обмене данными с точками *F* и *E*. Для решения этой проблемы могли бы быть проложены два дополнительных канала *F-R* и *R-E*. Однако, если число пользователей в пункте *R* невелико, то такое решение вряд ли может быть эффективным. Выход был найден путем разработки нового типа устройства, названного **мультиплексором ввода-вывода** (Add-Drop Multiplexer, **ADM**) (рис. 8.1, *б*). Мультиплексор ввода-вывода первичной сети имеет два магистральных порта и сравнительно небольшое количество портов для локально подключенных пользователей. Мультиплексор ввода-вывода позволяет *вывести* (операция Drop) из магистрального канала подканал, направив его данные на входной интерфейс локального пользователя, или *ввести* (операция Add) данные локального пользователя в магистральный канал.

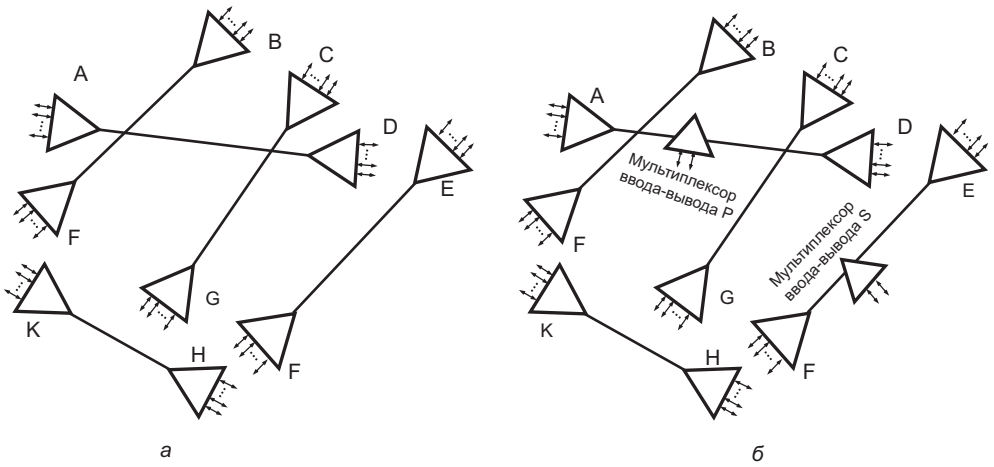


Рис. 8.1. Простейшие топологии первичных сетей

Со временем некоторые первичные сети разрастаются, у них увеличивается количество пользователей и соответственно растет число двухточечных линий связи. Это же происходит при слиянии нескольких первичных сетей. Топология сети приобретает сложный «запутанный» ячеистый вид (рис. 8.2, *а*). Возникает объективная необходимость связывания преимущественных, магистральных направлений между собой. Мультиплексоры ввода-вывода уже не решают эту проблему. Ключевым элементом в этих случаях становится **цифровой кросс-коннектор** (Digital Cross-Connect, DXC) — устройство, которое имеет несколько магистральных интерфейсов (а не один, как терминальный мультиплексор, или два, как мультиплексор ввода-вывода) и может выполнять коммутацию любых подканалов магистральных каналов с каналами пользователей или же коммутировать два любых магистральных канала между собой (рис. 8.2, *б*).

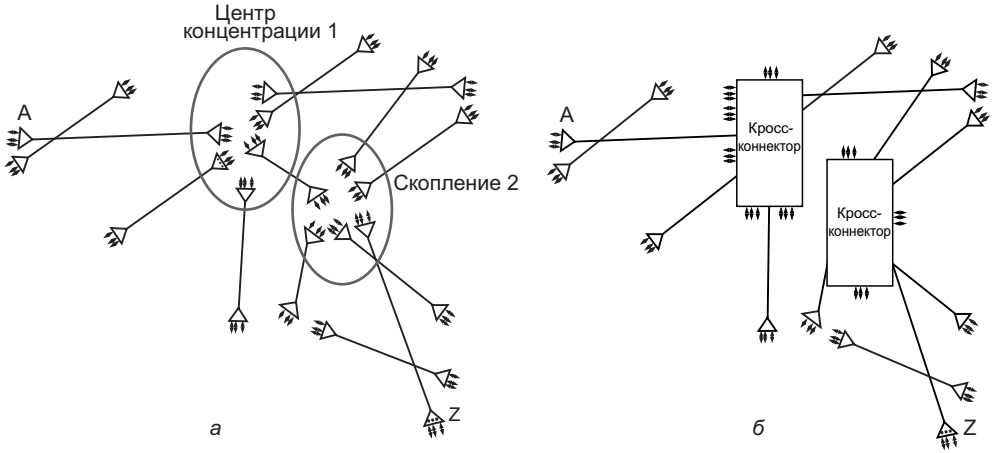


Рис. 8.2. Назначение кросс-коннекторов в первичных сетях

Кроме того, в состав первичной сети могут входить **регенераторы сигналов**, необходимые для преодоления ограничений по расстоянию между мультиплексорами. Эти ограничения зависят от мощности оптических передатчиков, чувствительности приемников и затухания волоконно-оптического кабеля. Регенератор преобразует оптический сигнал в электрический и обратно, при этом восстанавливается форма сигнала и его временные характеристики. Регенерацию сигналов могут выполнять также мультиплексоры.

Порты мультиплексора (рис. 8.3) делятся на агрегатные и трибутарные. **Трибутарные порты** часто называют также портами ввода-вывода, а **агрегатные** — линейными или магистральными портами. Эта терминология отражает типовые топологии первичных сетей, где имеется ярко выраженная магистраль в виде линейной цепи или кольца, по которой передаются потоки данных, поступающие от оборудования пользователей сети через трибутарные порты, втекающие в агрегированный поток («tributary» дословно означает «приток»).



Рис. 8.3. Простейшая первичная сеть

Статичность нагрузки. Иерархия скоростей

Еще одна особенность первичных сетей состоит в том, что нагрузка, которая поступает в сеть, не меняется в течение достаточно большого периода времени. Основные параметры этой нагрузки должны быть *заранее* известны администратору сети. Он должен знать скорость каждого из потоков, поступающих на каждый порт мультиплексора, являются ли эти данные синхронными или асинхронными, кому они предназначаются. На основании этих данных должны быть *сконфигурированы* все устройства сети, и после этого сеть будет работать в таком автоматическом режиме до тех пор, пока не возникнет потребность в изменении нагрузки. Эта задача упрощается за счет того, что виды нагрузки не являются произвольными. Исторически первичные сети специализируются на передаче синхронного трафика со строго определенными скоростями, кратными скорости элементарного канала DS0 64 Кбит/с, например 2 Мбит/с (30 каналов DS0).

Пользовательские потоки данных мультиплексируются в агрегатные потоки также определенной скорости, и скорости агрегатных потоков также образуют свою иерархию, обычно с постоянным коэффициентом кратности, когда скорость следующего уровня в N раз выше скорости предыдущего уровня. *Иерархия скоростей* с постоянным коэффициентом кратности облегчает организацию синхронного TDM-мультиплексирования потоков, так как всегда известно, сколько потоков одного уровня может быть мультиплексировано в поток другого уровня.

Первичная сеть работает на основе постоянной конфигурации своих мультиплексоров. Это означает, что в каждом мультиплексоре администратором сети создаются таблицы мультиплексирования и коммутации, которые определяют, как пользовательские потоки каждого трибутарного интерфейса без пауз и задержек мультиплексируются в агрегатные потоки, коммутируются на те или иные выходные порты. Эти таблицы учитывают скорости каждого потока, наличие свободных тайм-слотов в агрегатных потоках, в которые нужно мультиплексировать пользовательские потоки (если агрегатный поток рассчитан на мультиплексирование, например, 24 пользовательских потоков, то 25-й в него добавить уже невозможно),

и возможности коммутации потоков. После того как конфигурация каждого мультиплексора определена и согласована с конфигурацией остальных мультиплексоров сети, сеть может начать работу. И если конфигурации мультиплексоров были определены администратором сети правильно, то каждый поток данных пользователя будет перемещаться по сети и поступит к получателю с той же скоростью, с которой он поступил на входной порт.

Первичная сеть представляет довольно жесткую, статичную систему, работающую в соответствии с одной и той же конфигурацией долговременных каналов. В первичной сети не существует аналогов протоколов маршрутизации компьютерных сетей, которые позволяют автоматически динамично учитывать изменения, происходящие в сети, — добавление новых пользователей, изменение маршрутов прохождения данных в сети и т. п. Отсутствие протоколов маршрутизации связано с тем, что изменения конфигурации первичной сети происходят редко, так что добавление таких протоколов к функциям их мультиплексоров не считается целесообразным.

Первичная сеть не всегда работает с полной нагрузкой. Даже в том идеальном случае, когда каждый пользовательский поток постоянно передает байты, несущие полезную информацию (мы знаем, что для компьютерного трафика это невозможно, но возможно для телефонного трафика в течение сеанса связи), в сети, скорее всего, имеются агрегатные каналы, которые в некоторых своих тайм-слотах переносят «пустоту», то есть, по сути, эти слоты не используются. Это происходит из-за того, что сеть всегда проектируется с учетом развития, то есть не на вполне определенное количество пользователей с вполне определенными скоростями своего оборудования, а с некоторым запасом, позволяющим подключать новых пользователей без необходимости покупать и устанавливать новые мультиплексоры. Поэтому агрегатные каналы всегда имеют запас, например, позволяя переносить данные 2000 пользовательских потоков определенной скорости, в то время как у оператора связи имеется около 1500 пользователей. В результате около 500 тайм-слотов просто не используются (лотки конвейера приходят пустыми).

При *проектировании* первичной сети необходимо знать или уметь предвидеть исходные данные о том, в каких географических точках располагаются пользователи сети, какие скорости имеет пользовательское оборудование, между какими пользователями необходимо обеспечить соединения и какую скорость должны иметь эти соединения. Если исходные данные были неверны, то в результате может оказаться, что не все требуемые соединения можно выполнить из-за нехватки пропускной способности в некотором участке сети. Эта ситуация незнакома проектировщикам и администраторам компьютерных сетей, так как неверный выбор пропускной способности отдельных сегментов сети может привести к задержкам и потерям пакетов из-за переполнения очередей в буферах маршрутизаторов и коммутаторов, то есть к снижению качества соединений, но не к полному их отсутствию.

Функции мультиплексора

К основным функциям мультиплексора относятся: формирование кадров, отображение пользовательских данных в поле данных кадра, синхронизация (выравнивание скоростей) кадров, мультиплексирование и коммутация.

На рис. 8.4 показан типичный фрагмент первичной сети. Рассмотрим работу мультиплексора A. Он имеет два трибутарных порта и два агрегатных порта той же скорости. В его задачу входит передача пользовательских данных с трибутарного порта 1 на агрегатный порт 1, а данных с трибутарного порта 2 — на агрегатный порт 2.

Рассмотрим, как он выполняет эту задачу для трибутарного порта 1. Мультиплексор А принимает последовательность байтов пользователя в трибутарный порт, *образует из них кадры* и в виде кадров передает на соответствующий агрегатный порт.

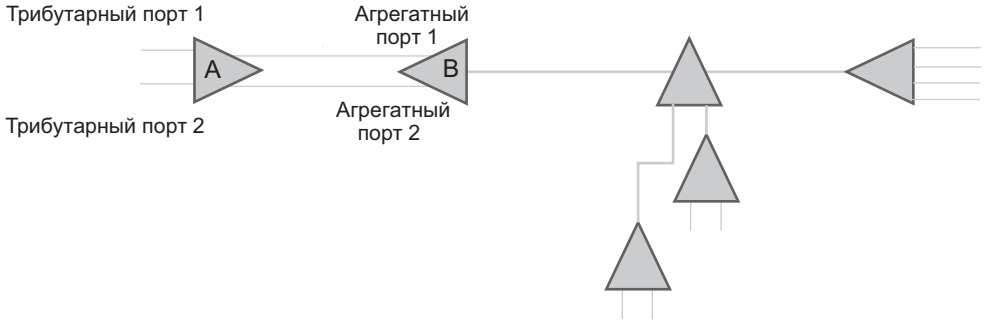


Рис. 8.4. Фрагмент первичной сети

Важно понимать, что понятие «кадр» в синхронной первичной сети *не совпадает* с понятиями «кадр» и «пакет» в сетях с коммутацией пакетов. Вспомним, что кадры/пакеты в пакетных сетях определяются как единицы данных, которые могут обрабатываться сетью *независимо* друг от друга, двигаясь при этом с разной скоростью и по разным маршрутам, буферизоваться, обгонять друг друга и т. д. Разделение данных на кадры в пакетных сетях преследует цель повышения *эффективности передачи неравномерного трафика.*

Совсем по-другому обстоят дела с кадрами, формируемыми в процессе синхронного разделения времени. Кадры синхронного TDM являются нарезкой непрерывного потока данных, передаваемого по составному каналу сети с коммутацией каналов. Они всегда имеют равную величину и следуют *синхронно,* без пауз друг за другом. В простейшем случае кадры могут состоять только из поля данных без какого-либо заголовка. Однако в развитых технологиях, использующих разделение времени (SDH и OTN), кадры TDM тоже снабжаются заголовком, который содержит вспомогательную информацию, необходимую для локализации и исправления ошибок, реконфигурирования сети, мониторинга качества работы сети, а также синхробайты, необходимые для распознавания начала кадра и его синхронизации. Таким образом, основной целью деления потока на кадры в технологии TDM является *эффективность управления сетью.*

Кадры первичных сетей имеют свои особенности не только по содержанию, но и по форме представления. Последовательность байтов кадра принято изображать в виде *матрицы*, так как заголовок кадра разбивается на порции, которые периодически вставляются между порциями данных, чтобы обеспечить более равномерное появление данных на выходе мультиплексора, а не задерживать их в буфере на все то время, которое требуется для полного прохождения заголовка кадра.

На рис. 8.5, *а* показана схема преобразования кадра традиционного формата, состоящего из поля заголовка и поля данных, в матрицу. Пусть кадр состоит из 20 байт, четыре из которых отведены под заголовок: синхробайта С, поля ИД идентификатора мультиплексора отправителя пользовательских данных и двух байтов контрольной суммы КС. Байты заголовка при передаче кадра через равные интервалы «вклиниваются» в поле данных, образуя кадр с расщепленным заголовком. Эта линейная структура преобразуется в матрицу, в которой первый

столбец содержит заголовок кадра. Заметим, что представление кадров с расщепленным заголовком в виде матрицы делается только для удобства понимания. В действительности же содержимое кадров выдается на линию связи *в виде последовательности битов*.



а) Схема представления кадра в виде матрицы



б) Расширение поля заголовка в мультикадре

Рис. 8.5. Особенности представления кадров первичных сетей

Важной особенностью кадров первичных сетей (см. рис. 8.5) является то, что значительная часть информации из заголовков характеризует не отдельный кадр, а поток кадров в целом. Поля заголовка могут содержать значения, которые не относятся к пользовательским данным, следующим непосредственно за заголовком. Это несколько странное обстоятельство связано с непрерывностью обработки потока байтов синхронным мультиплексором первичной сети без буферизации. Так, контрольная сумма пользовательских данных вычисляется последовательно, байт за байтом в темпе их поступления и становится известной только после приема всех байтов, образующих кадр, в нашем случае — после приема 20 очередных байтов. То есть к моменту вычисления контрольной суммы кадр уже закончился и должен начаться следующий. Поэтому мультиплексор помещает контрольную сумму данных кадра 1 в заголовок кадра 2, другого варианта у него нет.

Другой особенностью кадров первичных сетей является *иерархия* кадров, которая приводит к образованию **мультикадра**. Мультикадр состоит из нескольких последовательных кадров, в нашем примере — из трех. Мультикадр позволяет расширить некоторое поле заголовка кадра, добавив к нему такие же заголовки других кадров, а затем объявив их общими для всех кадров, входящих в мультикадр. Рассмотрим, например, поле ИД, в котором хранится идентификатор мультиплексора, который сгенерировал кадр. Предположим, что длина идентификатора 3 байта. Однако вместо того, чтобы отводить под идентификатор 3 байта в каждом кадре, его сделали равным одному байту и распределили это поле по полям ИД трех кадров мультикадра — первый байт поля ИД поместили в первый кадр мультикадра, второй — во второй, третий — в третий кадр. Информация, находящаяся в поле ИД, не меняется от мультикадра к мультикадру — она представляет собой часть конфигурации сети, которая, как замечено, меняется редко. Мультиплексоры сети, получающие кадры

от мультиплексора A, периодически считывают байты поля ИД и составляют полный идентификатор из двух последовательных байтов этого поля, полученных от двух кадров, входящих в мультикадр. Если бы идентификатор состоял не их двух, а из 20 байтов, то мультикадр нужно было бы составлять из 20 последовательных кадров.

Мультиплексирование потоков нужно для того, чтобы вместо нескольких агрегатных каналов, как это имеет место в примере между мультиплексорами A и B, можно было применить только один агрегатный канал, но работающий на более высокой скорости. Это позволяет экономить физические каналы связи. На рис. 8.4 агрегатные каналы с мультиплексированием используются между мультиплексорами B и C, C и D и др. Мультиплексирование выполняется в соответствии с техникой TDM-мультиплексирования.

Технологии PDH и SDH

Технологии PDH и SDH обладают всеми свойствами, которые были приведены в предыдущем разделе при описании первичных сетей: это сети с коммутацией каналов, с простейшей топологией двухточечных линий связи, с мультиплексорами в качестве основного вида оборудования.

Технология плезиохронной цифровой иерархии (PDH) была разработана в конце 60-х годов компанией AT&T для решения проблемы связи крупных коммутаторов телефонных сетей между собой. Начало было положено разработкой мультиплексора **T1**, который позволял в цифровом виде мультиплексировать, передавать и коммутировать (на постоянной основе) голосовой трафик 24 абонентов. Так как абоненты по-прежнему пользовались обычными телефонными аппаратами, то есть передача голоса шла в аналоговой форме, то мультиплексоры T1 сами осуществляли оцифровывание голоса с частотой 8000 Гц и кодировали голос методом импульсно-кодовой модуляции. Для разделения канала использовалось *мультиплексирование с разделением времени* (TDM). В кадре T1 последовательно передается по одному байту каждого абонента, а после 24 байтов вставляется один *бит синхронизации*, итого 24 × 8 + 1 = 193 бита за 125 мкс. В результате каждый абонентский канал образовывал цифровой поток данных 64 Кбит/с (DS0), а мультиплексор T1 обеспечивал его передачу на скорости 1,544 Мбит/с.

В качестве средств мультиплексирования при соединении крупных телефонных станций каналы T1 имели недостаточную пропускную способность, поэтому была реализована идея образования каналов с *иерархией скоростей*. Четыре канала типа T1 объединили путем побайтного временно́го мультиплексирования в канал следующего уровня цифровой иерархии — T2, передающий данные со скоростью 6,312 Мбит/с. Канал T3, образованный путем объединения семи каналов T2, имеет скорость 44,736 Мбит/с. Канал T4 объединяет шесть каналов T3, и в результате его скорость равна 274 Мбит/с. Описанная технология получила название **системы T-каналов**.

Технология систем T-каналов была стандартизована Американским национальным институтом стандартов (ANSI), а позже — международной организацией ITU-T. В результате внесенных ITU-T изменений возникла несовместимость американской и международной версий стандарта PDH. Аналогом систем T-каналов в международном стандарте являются каналы типа **E1**, **E2**, **E3** и **E4** с отличающимися скоростями — 2,048, 8,488, 34,368 и 139,264 Мбит/с соответственно. Канал E1 оперирует кадрами длиной в 32 байта, из которых 30 представляют байты 30 пользовательских потоков 64 Кбит/с, а два байта составляют заголовок кадра, в котором и переносится служебная информация.

При мультиплексировании нескольких пользовательских потоков в мультиплексорах PDH применяется техника, известная как **бит-стафинг**. К этой технике прибегают, когда скорость пользовательского потока оказывается несколько меньше, чем скорость объединенного потока. Подобные проблемы могут возникать в сети, состоящей из большого количества мультиплексоров, несмотря на все усилия по централизованной синхронизации узлов сети. В результате мультиплексор PDH периодически сталкивается с ситуацией, когда какой-либо из объединяемых потоков «опаздывает» и мультиплексору «не хватает» бита для представления этого потока в объединенном кадре. В этом случае мультиплексор просто вставляет в объединенный поток бит-вставку и отмечает этот факт в служебных битах объединенного кадра. При демультиплексировании объединенного потока бит-вставка удаляется из пользовательского потока, который возвращается в исходное состояние.

Отсутствие полной синхронности потоков данных при объединении низкоскоростных каналов в высокоскоростные и дало название технологии PDH: «плезиохронный» означает «почти синхронный».

Способ выравнивания скоростей потоков при их мультиплексировании с помощью бит-стафинга прост, но он приводит к сложностям при демультиплексировании, так как заранее не известно, на каком уровне (или уровнях) был добавлен бит. Например, чтобы получить данные абонентского канала 64 Кбит/с из кадров T3, нужно провести демультиплексирование этих кадров до уровня кадров T2, затем демультиплексировать кадры T2 до кадров T1, а уже затем выделить из них пользовательские потоки 64 Кбит/с.

Таким образом, в технологии PDH не могут быть использованы *мультиплексоры ввода-вывода и кросс-коннекторы*, так как эти устройства построены на возможности вывода отдельного подканала из магистрального канала без полного демультиплексирования последнего. Это сильно ограничивает возможности подключения новых пользователей к промежуточным устройствам.

Технология **синхронной цифровой иерархии SDH** (Synchronous Digital Hierarchy) была разработана в середине 80-х годов для создания высокоскоростных цифровых каналов, способных соединять крупные телефонные сети между собой. Разработчики технологии SDH создали гибкую иерархическую систему мультиплексирования, которая смогла поддерживать в качестве клиентских потоков как европейскую, так и североамериканскую систему каналов PDH. Магистраль сети SDH работает с пятью уровнями иерархии скоростей: STM-1 (Synchronous Transport Module level 1) — 155 Мбит/с, STM-4 — 622 Мбит/с, STM-16 — 2,4 Гбит, STM-64 — 10 Гбит/с и STM-256 — 40 Гбит/с. Коэффициент соотношения скоростей между смежными уровнями постоянен и равен 4, это упрощает схему мультиплексирования потоков.

В качестве среды передачи данных оборудование SDH использует *одномодовый волоконно-оптический кабель*, передавая по нему модулированные световые сигналы с длиной волны 1310 нм или 1550 нм.

Мультиплексоры SDH выполняют операции мультиплексирования и коммутации над электрическими сигналами — для этого они преобразуют *оптические* сигналы, пришедшие от оптических портов, в электрические, а после выполнения необходимых операций выполняют обратное преобразование сигналов. Такой тип работы называется работой по схеме **O-E-O**, от английского термина Optical-Electrical-Optical.

Топология сетей SDH достаточно разнообразна. Она представляет собой кольца, линейные цепи мультиплексоров и ячеистую топологию.

Кольцо SDH строится из мультиплексоров ввода-вывода, имеющих по крайней мере по два агрегатных порта (рис. 8.6, *а*). Пользовательские потоки вводятся в кольцо и выводятся из кольца через трибутарные порты, образуя двухточечные соединения (на рисунке показаны в качестве примера два таких соединения). Кольцо является классической регулярной топологией, обладающей потенциальной отказоустойчивостью — при однократном обрыве кабеля или выходе из строя мультиплексора соединение сохранится, если его направить по кольцу в противоположном направлении. Кольцо обычно строится на основе кабеля с двумя оптическими волокнами, но иногда для повышения надежности и пропускной способности применяют четыре волокна.

Линейная цепь (рис. 8.6, *б*) — это последовательность мультиплексоров ввода-вывода с двумя терминальными мультиплексорами на окончаниях линии. Обычно сеть с топологией цепи применяется в тех случаях, когда узлы имеют соответствующее географическое расположение, например вдоль магистрали железной дороги или трубопровода. Правда, в таких случаях может применяться и **плоское кольцо** (рис. 8.6, *в*), обеспечивающее более высокий уровень отказоустойчивости за счет двух дополнительных волокон в магистральном кабеле и по одному дополнительному агрегатному порту у терминальных мультиплексоров.

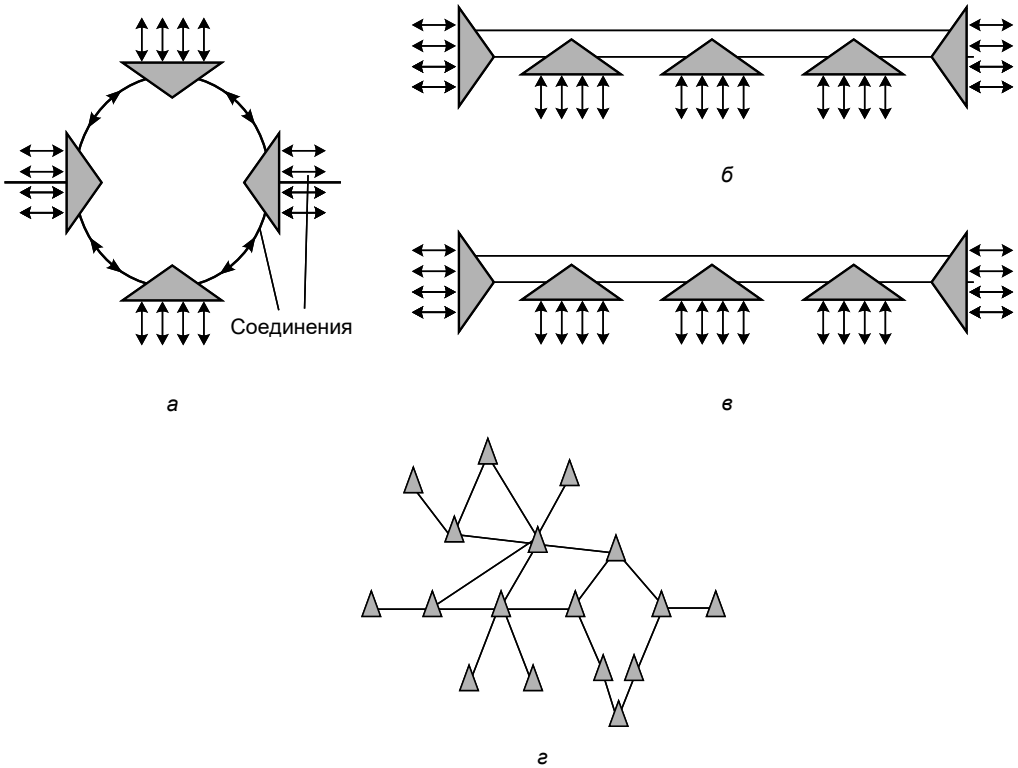


Рис. 8.6. Типовые топологии SDH

Наиболее общим случаем является **ячеистая топология** (рис. 8.6, *г*), при которой мультиплексоры соединяются друг с другом большим количеством агрегатных связей, за счет чего сеть может достичь очень высокой степени производительности и надежности.

Мультиплексирование модулей STM уровней *выше первого* выполняется в соответствии с простым правилом: каждый последующий уровень STM-N получается *мультиплексированием четырех блоков предыдущего уровня STM-(N – 1)*.

Например, содержимое кадра STM-4 представляет собой чередующиеся байты кадров STM-1, причем позиция байтов каждого из кадров STM-1 известна и фиксирована, как это и должно быть при *TDM-мультиплексировании*. Демультиплексирование может быть выполнено «на лету». Например, для вывода из кадра STM-4 третьего кадра STM-1 достаточно копировать в выходной порт получателя байты, находящиеся в позициях (таймслотах), кратных трем.

Время цикла, или, что одно и то же, время приема данных от всех мультиплексируемых потоков и формирования из них кадра, — всегда равно *125 мкс*[1], независимо от уровня скорости STM-N магистрального канала. Следовательно, размер кадра (включая заголовок) связан со скоростью следующим соотношением: (STM-N) × 125/8 байт.

Основой техники мультиплексирования SDH являются виртуальные контейнеры и указатели на них. **Виртуальный контейнер** является носителем пользовательских данных внутри кадров STM-1–STM-256. **Указатель** является целым числом, который определяет текущее положение виртуального контейнера в кадре. Механизм указателей — *ключевой* в технологии SDH, он заменяет принятое в PDH выравнивание скоростей асинхронных источников посредством бит-стафинга. С помощью указателя виртуальный контейнер может «смещаться» в определенных пределах внутри кадра, если скорость пользовательского потока несколько отличается от скорости кадра SDH, в который этот поток мультиплексируется.

Именно благодаря системе указателей мультиплексор находит положение пользовательских данных в синхронном потоке байтов кадров STM-N и «на лету» извлекает их оттуда, чего механизм мультиплексирования, применяемый в PDH, делать не позволяет.

Сеть SDH может выполнять коммутацию любого виртуального контейнера между любым входным и выходным интерфейсами сети, обеспечивая тем самым работу сетей с ячеистой топологией.

Отказоустойчивость была одной из главных целей разработчиков технологии SDH. Она обеспечивается за счет ряда механизмов, которые позволяют сети быстро (менее чем за 50 мс) восстановить работоспособность в случае отказа какого-либо элемента сети — линии связи, порта мультиплексора или мультиплексора в целом. Такое быстрое переключение на резервный путь очень важно при передаче голосового трафика, поскольку обеспечивает приемлемое для абонентов телефонной сети качество связи даже в условиях отказов элементов сети. Подробное описание технологий PDH и SDH вы можете найти на сайте книги.

(S) *Технология PDH*

(S) *Технология SDH*

---

[1] Выбор времени цикла равным интервалу между замерами оцифрованного с частотой 8 кГц голоса показывает ориентацию технологии SDH прежде всего на качественную передачу телефонного трафика.

Технология OTN

Причины создания сетей OTN

В конце 90-х годов все больше стали проявляться недостатки технологии SDH, связанные прежде всего с ее изначальной ориентацией только на голосовой трафик. Основные недостатки технологии SDH состоят в следующем:

1. *Мультиплексоры SDH оперируют слишком «мелкими» единицами коммутации.* Наличие таких клиентских каналов, как 1,5 Мбит/с, 2 Мбит/с или 34 Мбит/с, усложняет оборудование сети. Когда скорость агрегатного канала возрастает до нескольких десятков гигабит в секунду и канал переносит при этом сотни индивидуальных виртуальных контейнеров в своих кадрах, количество операций мультиплексирования и коммутации, которые нужно производить оборудованию SDH в единицу времени, становится настолько большим, что его процессорные модули перестают справляться с вычислительной нагрузкой. Примером технологии, которая пострадала от такой ситуации, является технология ATM (см. материалы на сайте книги). Ее коммутаторы справлялись с обработкой ячеек данных очень маленького размера в 53 байта, пока скорости передачи данных были меньше 1 Гбит/с. Барьер в 1 Гбит/с оборудование ATM преодолеть не смогло: для этого ей потребовались дорогие высокопроизводительные процессоры, в то время как коммутаторы Ethernet, оперирующие пакетами данных в 1500 байт, справлялись со своей работой, используя процессоры, обладающие существенно более низкой скоростью и стоимостью.
2. *Технология SDH не учитывает особенности трафика различного типа.* Разработчиками технологии SDH принимался во внимание только голосовой трафик, тогда как сегодня преобладающим является компьютерный трафик. Отображение потоков Ethernet со скоростью передачи данных 1 Гбит/с и 10 Гбит/с в кадры SDH возможно, но приводит к большим потерям пропускной способности, а потоки со скоростью передачи данных 100, 200, 400 Гбит/с вообще превосходят максимальную скорость SDH.
3. *Недостаточная интеграция с сетями DWDM.* Сети SDH создавались до появления технологии DWDM, поэтому в них применялось простое кодирование NRZ, имеющее *широкий спектр* сигнала, не рассчитанный на ширину спектральных каналов частотных планов DWDM. До скорости 10 Гбит/с спектр такой ширины был еще приемлем, но для более высоких скоростей потребовалось другое решение. Недостаточная интеграция с DWDM проявилась и в отсутствии стандарта на использование *кодов прямой коррекции ошибок* FEC (см. главу 7). Коды FEC позволяют не только обнаруживать ошибки, но и исправлять их. Это свойство очень полезно при увеличении количества спектральных каналов в оптическом волокне сети DWDM. Действительно, в результате роста числа спектральных каналов происходит сближение спектров сигналов соседних волн, а значит, увеличивается взаимное влияние этих сигналов и, как следствие, возрастают искажения сигналов и появляются битовые ошибки. В таких условиях появление эффективной процедуры FEC, позволяющей «на лету» устранять значительную часть этих ошибок, так что этими ошибками можно пренебречь, дает возможность увеличить количество спектральных каналов. В технологии SDH стандарт на использование кодов FEC долгое время отсутствовал. Когда же стандарт был, наконец, принят, оказалось, что он обладает недостаточной способностью снижать вероятность битовой ошибки.

4. *Необходимость централизованной синхронизации всей сети* усложняет сети SDH. В то же время при передаче компьютерного трафика такая централизованная синхронизация вообще не нужна, потому что достаточно обеспечить синхронизацию между передатчиком и приемником непосредственно соединенных портов.

Учитывая эти и другие недостатки SDH, было решено оставить попытки улучшения «здания» SDH, построенного на устаревшем фундаменте, и создать новую технологию **оптических транспортных сетей** (Optical Transport Network, **OTN**)[1], позаимствовав у технологии SDH несколько архитектурных принципов:

- иерархию скоростей с коэффициентом умножения 4 при переходе к более высокому уровню скорости;
- побайтное TDM-мультиплексирование кадров более низкого уровня при их передаче в поле данных кадров более высокого уровня;
- четырехуровневую функциональную структуру;
- типы оборудования: терминальный мультиплексор, мультиплексор ввода-вывода, коммутатор (кросс-коннектор), регенератор;
- топологии сети: линейная цепь без ввода-вывода в промежуточных точках, линейная цепь с вводом-выводом в промежуточных точках, кольцо, ячеистая топология.

В то же время при реализации каждого из этих элементов архитектуры разработчики технологии OTN вносили изменения с учетом описанных выше недостатков технологии SDH и требований времени.

Иерархия скоростей

На начальных этапах развития технологии OTN в конце 90-х — начале 2000-х годов ее разработчики еще не осознавали всю важность компьютерного трафика для телекоммуникационных сетей. Сети OTN позиционировались прежде всего как магистральные высокоскоростные сети для объединения сетей SDH, а клиенты с интерфейсами Ethernet не принимались во внимание. Поэтому в начальных версиях стандартов OTN было определено всего *три уровня скорости*, которым должны были удовлетворять потоки клиентов:

- уровень 1 со скоростью пользовательских данных 2,488 Гбит/с (STM-16);
- уровень 2 со скоростью 9,953 (STM-64);
- уровень 3 со скоростью 39,813 (STM-256, наивысшая скорость SDH).

В дальнейшем такое недальновидное решение было исправлено, и в перечень скоростей, допустимых для потоков данных клиентов, были добавлены скорости, соответствующие *иерархии скоростей Ethernet*, а также уровень «любая скорость» (табл. 8.1).

Таблица 8.1. Иерархия скоростей технологии OTN

Клиентский кадр	Битовая скорость клиента (Гбит/с)	Уровень скорости (индекс k)	Битовая скорость кадров OTN (Гбит/с)
Ethernet	1	0	–
STM-16	2,488	1	2,666

[1] Архитектура сетей OTN описана в стандарте ITU-T G.872, а наиболее важные технические аспекты работы узла сети OTN — в стандарте G.709.

Клиентский кадр	Битовая скорость клиента (Гбит/с)	Уровень скорости (индекс k)	Битовая скорость кадров OTN (Гбит/с)
STM-64 или Ethernet 10G	9,953 или 9,53	2	10,709
STM-256	39,813	3	43,018
Ethernet 100G Ethernet, MPLS, Fibre Channel	100 Любая	4	111,809 –
Ethernet 200G, 400G, $n \times 100$G	$n \times 100$	Cn[1]	$n \times 105{,}258$

Функциональные уровни OTN

В технологии OTN определено четыре функциональных уровня. Рассмотрим три из них, поскольку функции фотонного уровня SDH нам уже известны — их выполняет технология DWDM.

Уровень блока пользовательских данных оптического канала (Optical Channel Payload Unit, **OPU**) является самым верхним функциональным уровнем технологии OTN. Он ответственен за отображение пользовательских данных, то есть кадров STM SDH или Ethernet, в **блоки OPU**. Заголовок блока OPU OH (OverHead) содержит информацию о типе пользовательских данных, переносимых полем данных, а также информацию, позволяющую выровнять скорости пользовательских данных и блока OTN, передаваемого на выходной интерфейс оборудования OTN.

Уровень блока данных оптического канала (Optical Channel Data Unit, **ODU**), хотя и является более низким уровнем, так же, как и уровень OPU, отвечает за передачу данных между *конечными* узлами сети OTN. Единицей данных на этом уровне является **блок ODU**. Заголовок блока ODU OH несет данные, необходимые для механизмов мониторинга и администрирования соединения «из конца в конец», например данные о типе и местонахождении неисправности, о задержке сигналов, данные, необходимые для работы механизмов отказоустойчивости. В функции уровня ODU входит также мультиплексирование и демультиплексирование блоков. В этом аспекте уровень ODU OTN во многом аналогичен уровню мультиплексной секции SDH.

Уровень транспортного блока оптического канала (Optical Channel Transport Unit, **OTU**) работает между двумя *соседними* узлами сети OTN, которые поддерживают функции электрической регенерации оптического сигнала. Уровень OTU передает свою единицу данных — **блок OTU** — на фотонный уровень, непосредственно в спектральный канал DWDM. Основное назначение этого уровня — обнаружение и исправление ошибок с помощью кодов FEC. Заголовок блока OTU включает биты контроля поля данных по четности (они позволяют определить искажение битов, но не исправить их, как это делает код FEC), бит индикации ошибки, обнаруженной контролем по четности, а также некоторую другую служебную информацию. Перед заголовком OTU OH помещены несколько байтов выравнивания, которые, во-первых, определяют начало кадра, а во-вторых, дополняют первую строку до нужного количества байтов. Уровень OTU соответствует уровню регенераторной

[1] Это обозначение для скоростей, кратных 100 (Century) Гбит/с.

секции SDH. Блок OTU представляет собой законченный кадр OTN, который передается по одному из спектральных каналов DWDM. В дальнейшем мы будем использовать названия *блок OTU* и *кадр OTN* как синонимы.

Уровни ODU и OPU работают с *электрическими сигналами*, получая их от уровня OTU, который преобразует оптические сигналы DWDM в электрические, а также выполняет обратное преобразование, то есть работает по схеме O-E-O. Как и оборудование SDH, оборудование OTN выполняет все операции над электрическими сигналами, используя оптические сигналы только для передачи данных между мультиплексорами. Кодирование оптических сигналов выполняет фотонный уровень.

Формат кадра OTN

Пользовательские данные, поступившие на входной порт мультиплексора OTN, последовательно обрабатываются средствами разных уровней, начиная с самого высокого уровня OPU (рис. 8.6, *а*). Каждый очередной уровень генерирует дополнительную служебную информацию, которая добавляется к блоку пользовательских данных в виде заголовков, превращая его в соответствующий данному уровню блок (модуль, кадр).

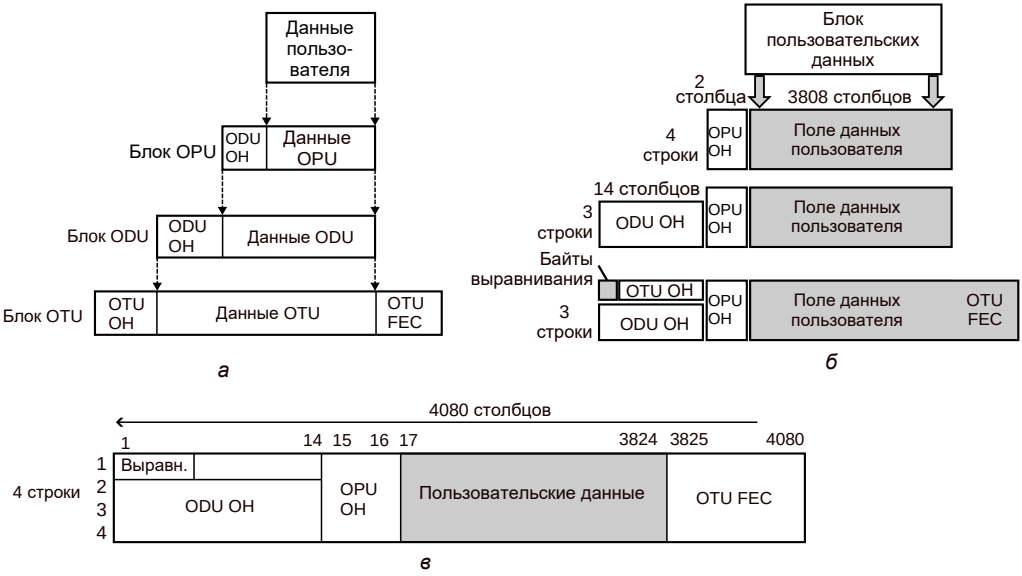


Рис. 8.7. Форматы кадра OTN

Представление кадра OTN в виде матрицы вносит некоторую специфику в описание этого процесса, но не меняет его сущности. На рис. 8.7, *б* показано матричное представление блока данных пользователя, размещенного в поле данных блока OPU, который обрастает заголовками всех трех уровней — OPU OH, ODU OH, OTU OH — и концевиком, содержащим код FEC.

Кадр OTN обычно представляют в виде матрицы, состоящей из 4080 столбцов-байтов и 4 строк (рис. 8.7, *в*). Данные пользователя располагаются с 17-го по 3824-й столбец

кадра и имеют длину 15 232 байта. Заголовок OPU OH занимает столбцы 15 и 16, а заголовок ODU OH — с 1-го по 14-й столбец во 2, 3 и 4-й строках. Заголовок OTU и байты выравнивания располагаются в первых 14 байтах первой строки. Последние 156 столбцов занимает концевик FEC.

Процедура прямой коррекции ошибок FEC в OTN основана на кодах **Рида — Соломона (RS)**, обозначаемых как RS (255, 239). Два параметра в скобках отражают тот факт, что данные в этом самокорректирующемся коде кодируются блоками по 255 байт, из которых 239 байт являются пользовательскими, а 16 байт представляют собой корректирующий код. Коды Рида — Соломона позволяют исправлять до восьми ошибочных байтов в блоке из 255 байт, что является очень хорошей характеристикой для самокорректирующегося кода. Применение кода Рида — Соломона позволяет улучшить отношение мощности сигнала к мощности шума на 5 дБ при уменьшении уровня битовых ошибок с 10^{-3} (без применения FEC) до 10^{-12} (после применения FEC). Этот эффект дает возможность увеличить расстояние между регенераторами сети на 20 км или же использовать менее мощные передатчики сигнала.

В технологии OTN кадр всегда имеет *фиксированный* размер — даже в случаях, когда он является результатом мультиплексирования нескольких таких же кадров (о том, как такое может произойти, см. далее). Учитывая, что в сетях OTN, как и в PDH/SDH, данные могут передаваться по агрегатным каналам с разными скоростями, время передачи кадра OTN является *переменным*. Например, кадр OTN первого уровня скоростной иерархии передается за 48 мкс, кадр второго уровня иерархии — за 12 мкс, а третьего — за 3 мкс.

Эти два свойства — фиксированный размер кадра и переменное время передачи кадра — принципиально отличают архитектуру OTN от архитектуры SDH, у которой размер кадра каждого последующего уровня STM в четыре раза больше размера кадра предыдущего уровня STM, а время передачи кадра по агрегатному каналу всегда равно точно 125 мкс.

Пример-аналогия

В городе N был объявлен тендер на разработку транспортной системы, разные участки которой характеризуются разной интенсивностью пассажиропотоков. В тендере участвовали две компании: SDH и OTN. SDH предложила ввести регулярное расписание движения на всех линиях — строго каждые 12,5 минуты от каждой остановки отправляется автобус, для учета различий в интенсивности потока пассажиров использовать на участках с малой интенсивностью потока пассажиров мини-автобусы, а на направлениях с интенсивным движением — большие двухэтажные автобусы. Другое решение предложила компания OTN: для всех перевозок используется один и тот же тип автобуса, но на более загруженных направлениях эти автобусы будут отправляться чаще. Как вы думаете, к какому выводу пришла комиссия?

Уровень скорости, на котором используется тот или иной блок, индицируется как индекс *k* в названии блока OPUk, например, блоки первого уровня иерархии скоростей — OPU1, ODU1, OTU1, блоки второго уровня — OPU2, ODU2, OTU2 и т. д.

Скорости различаются и у блоков, относящихся к *одному и тому же уровню иерархии* скоростей, например OPU1, ODU1, OTU1. Действительно, после того как данные пользователя поступают в сеть, к ним последовательно добавляется дополнительная информация в виде заголовков OPU OH, ODU OH и OTU OH, а также концевика OTU FEC. При каждом таком добавлении, чтобы сохранить неизменной скорость блока данных пользователя, сеть должна передавать вновь образовавшийся блок быстрее. Отсюда следует, что

скорости блоков данных разного типа (OPU, ODU и OTU), которые принадлежат одному скоростному уровню, но имеют разный объем заголовков, отличаются. Например, для первого уровня иерархии скоростей ($k = 1$) OPU1 равна 2,488 Гбит/с, ODU1 — 2,498 Гбит/с и OTU1 — 2,666 Гбит/с. Каждый последующий тип блока имеет большее число заголовков, а значит, скорость его передачи растет. При этом скорость пользовательских данных, отображенных в этих блоках, должна оставаться *неизменной*.

Отображение и выравнивание пользовательских данных

Основной идеей разработчиков технологии OTN было обеспечение *прозрачности* пользовательских данных, которая проявляется в том, что независимо от того, какой тип данных — SDH, Ethernet или данные других протоколов компьютерных сетей — переносит кадр OTN, вся специфика его обработки должна сказываться только на уровне OPU. На этом уровне данные пользователя должны отображаться в поле данных кадра OTN в соответствии с тем, к какому типу относятся данные пользователя. Если необходимо, скорости поступления пользовательских данных и передачи кадра OTN выравниваются.

Несмотря на то что иерархия скоростей первого поколения стандартов OTN ориентировалась исключительно на скорости клиентских потоков SDH, разработчики OTN предполагали, что клиентами сети будут как клиенты с оборудованием SDH, так и клиенты с оборудованием компьютерных сетей, представленным в основном интерфейсами Ethernet.

Одним из способов переноса данных Ethernet через сеть OTN является вариант, требующий *предварительной упаковки* кадров Ethernet в кадры STM SDH. В этом случае никакой специфической процедуры обработки трафика Ethernet не требуется.

Однако разработчики технологии OTN хотели обеспечить возможность переноса кадров Ethernet не только упакованными в кадры STM, но и непосредственно. Для этого необходимо было учесть *специфику пользовательских данных*, которые отличались не только скоростью, но и синхронностью — данные SDH представляют собой непрерывный синхронный поток байтов, а данные компьютерных сетей разделены на пакеты, прибывающие асинхронно.

В результате в технологии OTN имеется два типа процедур отображения пользовательских данных:

- процедуры отображения синхронного трафика, которые применяются для трафика SDH, в том числе кадров Ethernet, упакованных в кадры STM, а также других синхронных протоколов, например протоколов сетей хранения данных Fibre Channel (они в данной книге не изучаются);
- процедуры отображения асинхронного трафика компьютерных сетей.

Компьютерный трафик имеет, как мы знаем, пульсирующий характер. Хотя данные внутри каждого кадра (например, кадра Ethernet) поступают с постоянной битовой скоростью, из-за случайных пауз между кадрами средняя скорость поступления данных в мультиплексор OTN может колебаться в больших пределах, от нуля на некоторых периодах (периоды молчания) до битовой скорости протокола на других.

Для выравнивания такой неравномерности процедура отображения компьютерных данных вставляет в периоды молчания *пустые компьютерные кадры* — кадры, состоящие только из заголовка, в котором указывается нулевая длина поля данных.

Для того чтобы эта процедура была прозрачной, то есть не зависела от протокола компьютерной сети, исходный компьютерный кадр упаковывается в кадр формата *обобщенной процедуры инкапсуляции данных* **GFP** (Generic Framing Procedure). Этот формат кадра специально разработан для единообразного обрамления пользовательских кадров любого формата. Так, если данные поступают в виде кадров Ethernet, то перед размещением их в поле данных к ним добавляют заголовок GFP.

Заголовок GFP состоит из четырех байтов, два из которых отводятся для хранения длины его поля данных (ноль, если это пустой кадр, и длина инкапсулированного кадра в противном случае), а еще два — для контрольной суммы поля данных.

Процедура GFP поддерживает два режима работы: **GFP-F** (кадровый режим, или Frame Mode) и **GFP-T** (прозрачный режим, или Transparent Mode). Режим GFP-F предназначен для инкапсуляции компьютерных кадров, а режим GFP-T — для инкапсуляции данных синхронного чувствительного к задержкам трафика, отличного от SDH, например протокола сетей хранения данных Fibre Channel. Особенностью режима GFP-T является то, что исходный синхронный поток байтов разбивается на блоки равной длины и к ним добавляется заголовок, который позволяет распознать начало блока и корректность данных в нем по контрольной сумме блока. Заголовки кадров режимов GFP-F и GFP-T имеют одинаковый формат.

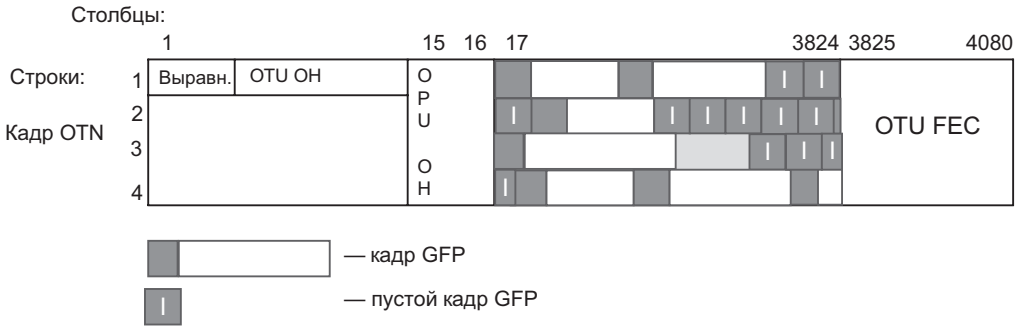


Рис. 8.8. Размещение кадров STM в полях данных кадров OTN

Итак, пусть на входной порт мультиплексора OTN поступает очередной кадр компьютерной сети. Перед передачей его в составе данных поля OPU в сеть он полностью буферизуется, так как асинхронный характер компьютерного трафика это позволяет; для него вычисляется контрольная сумма, добавляется заголовок GFP. Упакованный таким образом компьютерный кадр размещается в поле данных кадра OTN (рис. 8.8) и побайтно передается на выходной порт мультиплексора OTN. Если к моменту окончания передачи всех байтов этого компьютерного кадра следующий кадр компьютерной сети еще не поступил в буфер, то в поле данных кадра OTN помещается *пустой кадр GFP*, то есть четырехбайтный заголовок с указанием нулевой длины поля данных. Тем самым выравниваются скорости поступления пользовательских данных и передачи кадра OTN.

Мультиплексирование блоков OTN

Как и во всех других технологиях, мультиплексирование в OTN используется для эффективной передачи по высокоскоростной магистрали многих пользовательских потоков, имеющих более низкую скорость.

В OTN данные нескольких блоков ODU некоторого уровня скорости мультиплексируются в поле данных OPU более высокого уровня скорости, причем не обязательно следующего. Это означает, что блоки ODU1 могут мультиплексироваться как в блоки второго уровня OPU2, так и в блоки третьего уровня OPU3.

Принцип мультиплексирования

Рассмотрим технику мультиплексирования OTN на примере мультиплексирования четырех блоков ODU1 в один блок OTU2. Чтобы скорость данных блока низшего уровня осталась прежней в том случае, когда они переносятся по сети в блоках более высокого уровня, *кратность мультиплексирования должна соответствовать кратности скоростей уровней*. Так как скорость ODU2 в 4 раза выше скорости ODU1, то и кратность мультиплексирования ODU1 в ODU2 должна быть равна 4.

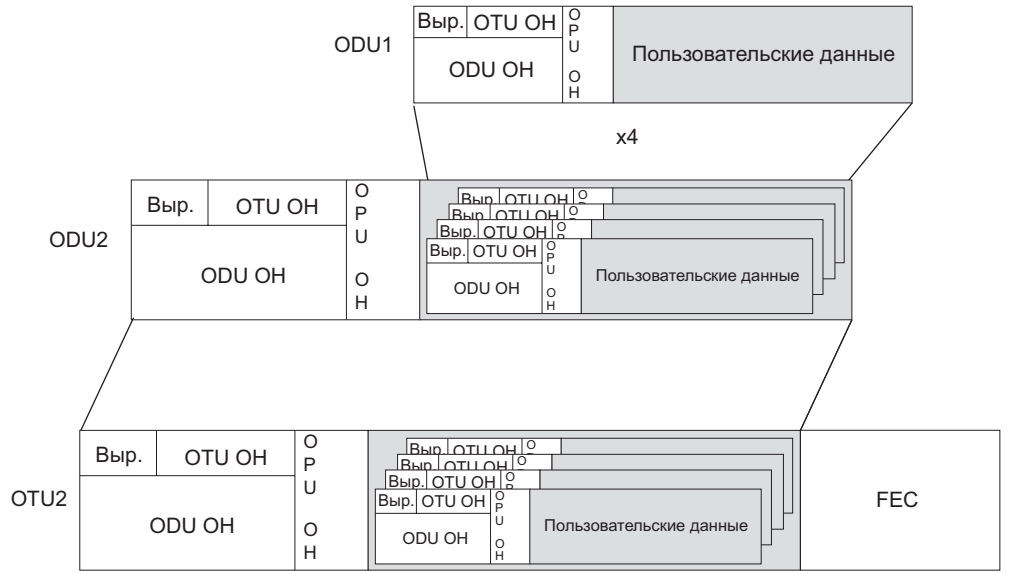


Рис. 8.9. Последовательность вложений блоков при мультиплексировании ODU1 в ODU2

На рис. 8.9 показано, что каждый из четырех блоков ODU1 мультиплексируется в поле данных ODU2, который после добавления концевика FEC превращается в блок OTU2. На рисунке этапы мультиплексирования блоков ODU1 показаны очень упрощенно, многие детали опущены, например, то, что для помещения четырех блоков ODU в поле данных ODU2 нужен *не один*, а четыре последовательных кадра ODU2.

Для упорядоченного размещения блоков ODU1 в поле OPU2 это поле разбивается на так называемые **трибутарные слоты** (Tributary Slot, TS). Каждый трибутарный слот содержит данные *одного* из мультиплексированных блоков.

На рис. 8.10 поле данных блока ODU2 разделено на четыре трибутарных слота — TribSlot1, TribSlot2, TribSlot3 и TribSlot4. Данные трибутарных слотов побайтно мультиплексированы, так что байты трибутарного слота 1 занимают позиции 17, 21, 25, ...; данные трибутарного слота 2 — позиции 18, 22, 26... и т. д. Таким образом, можно сказать, что трибутарный слот занимает несколько столбцов в матрице кадра ODU2, при этом столбцы разных трибутарных слотов чередуются.

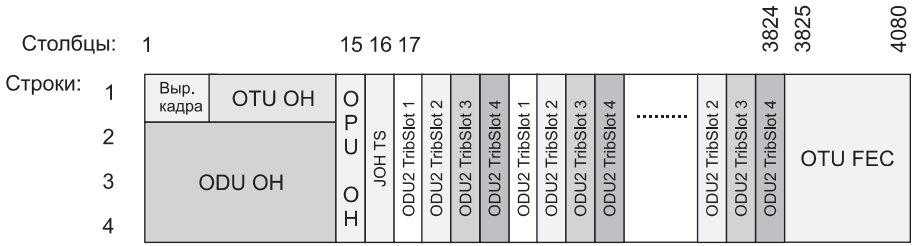


Рис. 8.10. Мультиплексирование блоков ODU1 в блок ODU2

Поскольку каждый блок ODU1 при мультиплексировании в блок ODU2 получает только четверть из полного размера поля данных ODU2, скорость передачи информации блока ODU1 в четыре раза ниже скорости передачи информации блока ODU2, как того и требует иерархия скоростей OTN.

Главной характеристикой трибутарного слота является его *скорость*. Трибутарные слоты одной скорости могут иметь разный размер в байтах в зависимости от того, в блок какого уровня они вложены. Так, в блоке ODU2, разделенном на 4 трибутарных слота, размер каждого слота скорости 2,5 Гбит/с равен 3808 байт, а в блоке ODU3, разделенном на 16 слотов, размер слота этой же скорости 2,5 Гбит/с уже в 4 раза меньше — всего 952 байта.

За счет выделения пользовательскому потоку различного числа трибутарных слотов можно обеспечить передачу данных разных потоков с разной скоростью. Так, если потоку выделено 3 трибутарных слота 2,5 Гбит/с, то он будет переноситься со скоростью 7,5 Гбит/с.

Мультиплексирование блоков ODU1 или ODU2 в поле данных блока ODU3 происходит аналогичным образом. В 16 тайм-слотов может быть помещены данные 16 блоков ODU1, или 4 блока ODU2, или их *любая комбинация*.

Описание мультиплексной структуры

Для того чтобы принимающий мультиплексор OTN мог понять, какие трибутарные слоты ODUk выделены для соответствующих блоков ODU более низкого уровня, в заголовке OPUk OH имеется поле **индикатора структуры данных** (Payload Structure Indicator, **PSI**). Рассмотрим, каким образом этот индикатор описывает выделение трибутарных слотов на примере блока ODU3, в слотах которого содержатся как блоки ODU1, так и ODU2.

Индикатор PSI может иметь длину 1 байт или 256 байт. Первый байт индикатора PSI называется полем типа нагрузки **PT** (Payload Type) и находится в 15-м байте 4-й строки кадра. По значению этого поля можно установить, находятся в поле данных блока OPU данные пользователя или же мультиплексированные кадры ODU.

Если значение PT равно коду, зарезервированному для пользовательских данных определенного типа (например, данных SDH или Ethernet), то это значит, что в поле данных нет

мультиплексированных кадров, поле данных OPU не разделено на трибутарные слоты и у него нет мультиплексной структуры. В этом случае PT является единственным байтом индикатора PSI.

Если же значение поля PT равно коду, зарезервированному для мультиплексированной нагрузки, то анализируются остальные 255 байт индикатора PSI. Эти байты также находятся в 15-м байте 4-й строки кадра, но они распределены по последовательным заголовкам кадров, входящих в мультикадр, то есть здесь использован прием расширения поля заголовка за счет объединения кадров в мультикадр.

Байты индикатора PSI последовательно описывают назначение каждого последующего трибутарного слота, так что позиция байта однозначно соответствует номеру трибутарного слота. Каждый байт-описатель трибутарного слота состоит из двух элементов. Первый показывает, блоку какого *уровня* принадлежит слот — ODU1 или ODU2 для нашего примера. Второй элемент содержит *номер трибутарного порта* (не путать с трибутарным слотом) мультиплексора, которому принадлежат эти данные. Таким образом, каждый байт-описатель индикатора PSI задает соответствие между номером трибутарного слота, уровнем блока ODU, помещенного в трибутарный слот, и номером трибутарного порта, от которого получен блок ODU.

Такая схема мультиплексирования является очень гибкой. Действительно, каждый трибутарный слот может быть назначен любому блоку ODU любого более низкого уровня, полученного от любого трибутарного порта. Информация о структуре мультиплексированных данных, содержащаяся в индикаторе PSI, используется оборудованием OTN как для *демультиплексирования*, так и для *коммутации* пользовательских данных, так как они точно указывают, в каком трибутарном слоте находятся данные определенного пользователя.

Общая схема мультиплексирования

Мы рассмотрели принципы мультиплексирования технологии OTN на примере трех уровней скоростей. Эти принципы остаются неизменными и при мультиплексировании других уровней скоростей, которые постепенно добавлялись к трем начальным уровням, ODU1 — ODU2 — ODU3. Общая схема мультиплексирования технологии OTN, отражающая все стандартизованные на сегодня уровни скорости как клиентов, так и кадров OTN, показана на рис. 8.11.

Возможности мультиплексирования некоторого блока ODUk (или блока ODUFlex) на схеме условно показаны пунктирными стрелками двух типов:

- стрелки, которые выходят из блока ODUk, показывают, в блоки каких типов ODUm этот блок ODUk может быть мультиплексирован;
- стрелки, входящие в блок OPUm, показывают, какие типы блоков ODUk могут быть мультиплексированы в данный блок OPUm.

Блоки OTUk помещаются в спектральные каналы DWDM, обозначенные на рисунке как OCh1...OChn.

Схема отражает следующие изменения в первоначальном варианте иерархии скоростей:

- *Для более эффективного мультиплексирования данных 1G Ethernet был определен формат ODU0 с битовой скоростью 1,25 Гбит/с.* До введения этого формата всегда нужно было объединять два потока Ethernet 1G и упаковывать их в кадры ODU1, что лишало операторов сети OTN необходимой гибкости.

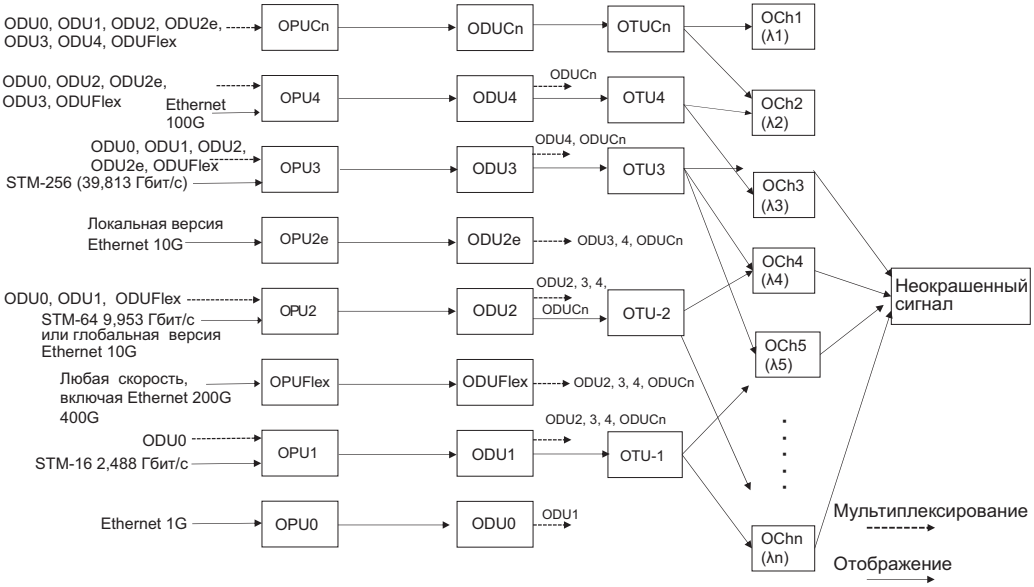


Рис. 8.11. Схема мультиплексирования технологии OTN

- *Появление стандарта 100G Ethernet вызвало стандартизацию нового, 4-го уровня скорости OTN с форматом кадра ODU4 и битовой скоростью кадра OTU4 111,8 Гбит/с, достаточной для переноса кадров Ethernet скорости 100 Гбит/с.* Этот уровень скорости потребовал применения более сложных методов кодирования, так как простое кодирование OOK/NRZ, применяемое в SDH, имеет слишком широкий спектр сигнала для наиболее применяемого на практике частотного плана DWDM с шагом 50 ГГц. Основным кодом для скорости 100 Гбит/с стал код PM-QPSK (Polarization Multiplexing — поляризационное мультиплексирование, Quadrature Phase Shift Keying — квадратурная фазовая манипуляция). В каждом такте кода PM-QPSK передается 4 бита дискретной информации. Ширина спектра такого сигнала позволяет передавать его в сетке 50 ГГц. Для надежного распознавания сигнала 100 Гбит/с в приемниках стали применяться цифровые сигнальные процессоры, которые программным путем распознают сигналы кода PM-QPSK на фоне помех, а также могут компенсировать линейные искажения сигналов из-за хроматической дисперсии.
- *Для кадров скоростей 200 и 400 Гбит/с, стандартизованных для Ethernet в 2017 году, введен формат кадра ODUCn,* с помощью которого можно переносить данные кадров Ethernet любой скорости, кратной 100 Гбит/с. Кадры ODUCn оперируют только трибутарными слотами скорости 5 Гбит/с. Для передачи данных со скоростями свыше 100 Гбит/с применяются два подхода. Первый подход основан на использовании более мощных методов кодирования светового сигнала — например, поляризационного мультиплексирования с квадратурной амплитудной модуляцией PM-16QAM[1] (256 состояний сигнала) и PM-32QAM (1024 состояния сигнала). Эти коды позволяют сузить спектр сигнала, но тем не менее даже при таких методах кодирования скорость

[1] Модуляция QAM рассмотрена в главе 7.

200 Гбит/с оказывается предельной для шага частотного плана в 50 ГГц, в связи с чем для скоростей 400 Гбит/с и выше приходится пользоваться более широким шагом, предлагаемым гибким частотным планом DWDM, например шагом 62,5 ГГц. Второй подход основан на идее суперканалов, когда для переноса данных используется несколько соседних волн частотного плана DWDM.

- *Введен формат кадра ODU2e с битовой скоростью для передачи кадров 10G Ethernet локальных сетей.* Необходимость в этой «чуть-чуть» увеличенной скорости ODU2 возникла из-за того, что существуют две версии стандарта Ethernet для скорости 10 Гбит/с — для локальных сетей и глобальных сетей. Версия Ethernet 10G для глобальных сетей была разработана специально для передачи ее кадров в кадрах STM-64 и поэтому имеет битовую скорость 9,953 Гбит/с. Локальная версия Ethernet 10G передает данные с битовой скоростью 10 Гбит/с, поэтому в кадрах ODU2 передаваться не может. Локальная и глобальная версии Ethernet несовместимы из-за разницы в битовой скорости, поэтому соединение локальных сетей с помощью сети OTN являлось проблемой. Для ее решения и был введен формат кадра ODU2e. Для блока ODU2e нет соответствующего блока OTU3e: он передается в блоках ODU3, чтобы не создать на магистрали сети две близкие скорости — OTU2 и OTU2e.
- *Для трафика со скоростью, значительно отличающейся от скоростей из иерархии OTN, введен формат кадра ODUFlex.* Данные помещаются в блок ODUFlex с той скоростью, с которой они поступают, а затем с помощью новой процедуры отображения *GMP* (*Generic Mapping Procedure*) отображаются в трибутарные слоты одного из блоков ODU верхнего порядка. Процедура GMP является усовершенствованным вариантом AMP и использует не один, а произвольное количество байтов в поле данных кадра OTN в качестве заполнителей, когда скорости пользовательских данных не хватает для заполнения всех байтов поля данных кадра OTN.

Организация сетей OTN

Сети OTN первого поколения имели достаточно простую архитектуру. Они строились на основе транспондеров DWDM, которые, наряду с операцией преобразования длин волн, выполняли и операцию отображения пользовательских данных в кадры OTN, а также модулирование цветной волны DWDM в соответствии с требуемой скоростью. В первых сетях OTN (рис. 8.12) не выполнялись ни мультиплексирование, ни коммутация — каждая волна DWDM переносила данные только одного пользователя. В этой связи технология OTN получила название *«цифровой оболочки»* DWDM, поскольку выполняла только функцию передачи цифровой информации — функцию, которую аналоговая технология DWDM выполнять не способна. Коммутация пользователей происходила в оборудовании SDH или в маршрутизаторах компьютерных сетей, а сеть OTN предоставляла этим пользователям только каналы «точка-точка».

Следующим этапом стало наделение блока транспондеров OTN/DWDM функцией мультиплексирования. Появились так называемые *макспондеры* (muxponder), способные объединить несколько пользовательских портов скорости ODU1 или ODU2 в один более скоростной выходной порт формата OTU3, а затем выполнить модуляцию одной из цветных волн частотного плана DWDM (рис. 8.13).

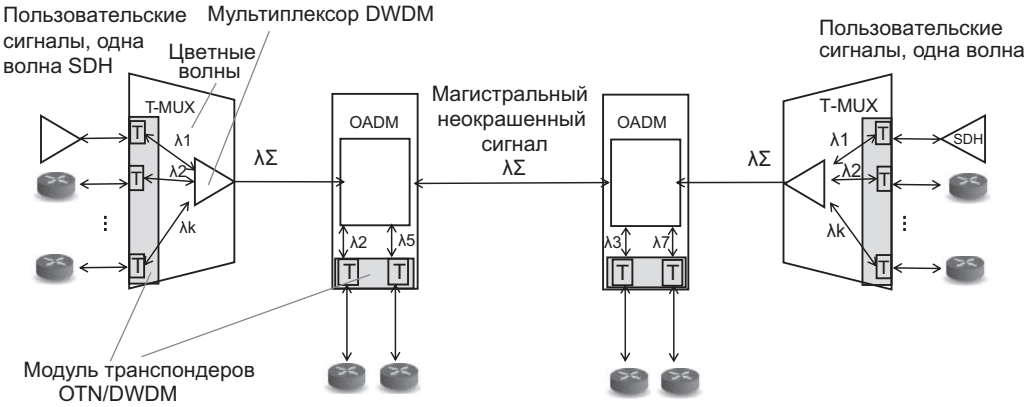


Рис. 8.12. Сеть OTN/DWDM с транспондерами

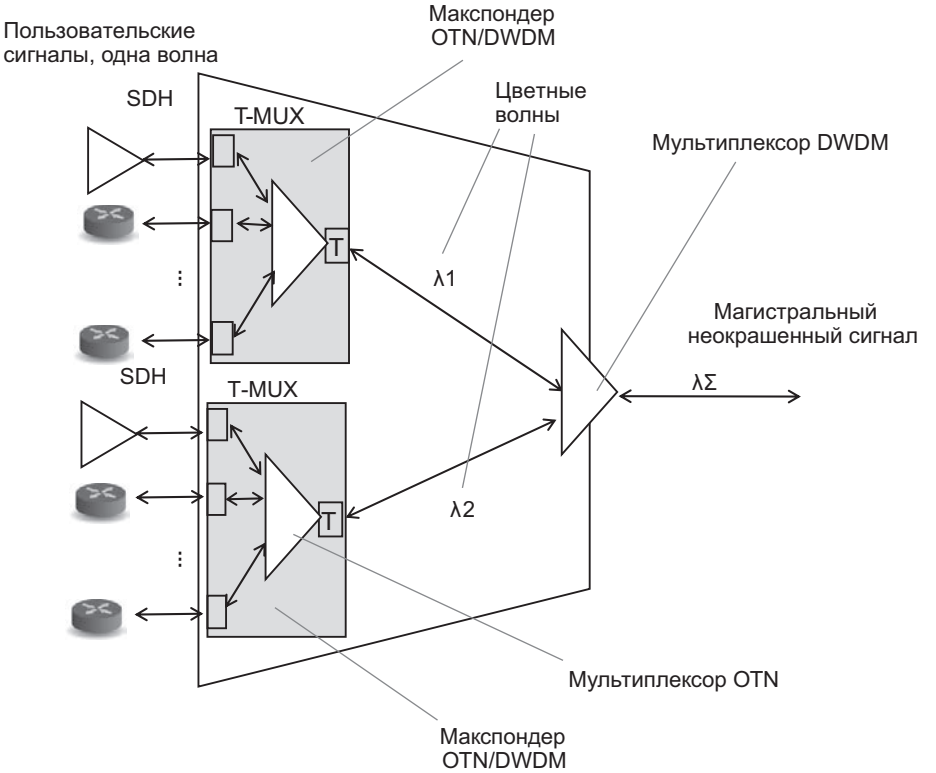


Рис. 8.13. Сеть OTN/DWDM с макспондерами

Модуль транспондеров включал мультиплексор OTN и транспондер DWDM. Сеть OTN/DWDM, построенная с использованием макспондеров, намного более экономно расходовала волны магистрального канала DWDM. Такой вариант сети позволял реализовать схему «оператор операторов», в соответствии с которой сеть телекоммуникационного оператора более высокого уровня служит магистралью сетей операторов нижних уровней. Тем не менее сервис OTN по-прежнему представлял собой набор каналов «точка-точка», и коммутация пользователей OTN средствами OTN не выполнялась.

И только примерно с середины 2010-х годов начало появляться оборудование OTN, выполняющее функции кросс-коннектора, то есть позволяющее коммутировать данные пользователей как между трибутарными, так и агрегатными портами (эта техника получила название OTN switching).

На рис. 8.14 показана такая сеть с пользователями, представленными оборудованием компьютерных сетей и сетей SDH. Здесь кросс-коннекторы OTN соединены друг с другом с помощью спектральных каналов, образованных мультиплексорами ROADM DWDM. Главное достоинство такой сети — в том, что в ней возможны транзитные соединения между любыми пользователями сети OTN, а не только между теми, мультиплексоры OTN которых непосредственно соединены спектральным каналом DWDM (одно из таких соединений показано на рисунке пунктирной линией).

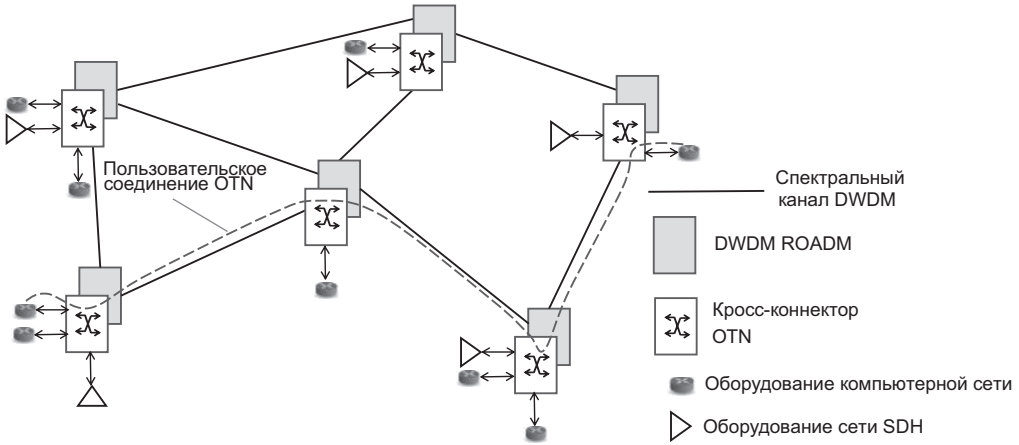


Рис. 8.14. Коммутируемая сеть OTN/DWDM

ГЛАВА 9 Технологии первичных сетей DWDM

Принцип работы

Технология **уплотненного волнового мультиплексирования** (Dense Wave Division Multiplexing, **DWDM**) предназначена для создания первичных сетей на основе оптических магистралей, работающих на мультигигабитных и терабитных скоростях.

До появления технологии DWDM все оборудование (как компьютерных, так и первичных сетей), применяющее волоконно-оптические кабели в качестве среды передачи информации, использовало модулированный сигнал *одной длины волны* — либо 850 нм из первого окна прозрачности при передаче по многомодовому волокну, либо 1310 нм или 1550 нм при передаче на большие расстояния по одномодовому волокну. Технология DWDM использует одновременно модулированные сигналы *нескольких длин волн*, увеличивая тем самым пропускную способность линии связи в то количество раз, сколько волн используется. За счет этого классического приема — применения большого числа параллельных носителей информации — обеспечивается революционный скачок в производительности сети.

Подобно кабелям с большим количеством параллельных жил здесь в одном оптическом волокне одновременно используется большое количество световых волн *разной длины* — **лямбд**. Этот термин возник в связи с традиционным для физики обозначением длины волны символом λ. Известно, что белый свет, воспринимаемый человеческим глазом, состоит из электромагнитных волн различной длины, образующих непрерывный спектр от примерно 400 нм до 780 нм. Этот факт демонстрируется существованием радуги на небе после дождя, а также простым опытом с оптической призмой, которая разлагает белый свет на его составляющие, воспринимаемые глазом как красные, оранжевые, желтые, зеленые, голубые, синие и фиолетовые лучи.

Оборудование DWDM поступает аналогично призме, расщепляя (демультиплексируя) составной сигнал, который называют **неокрашенным** (colourless), на отдельные составляющие сигналы — волны определенной длины, которые называют **окрашенными** (coloured). Отличие заключается в том, что технология DWDM работает с невидимыми для человеческого глаза инфракрасными волнами из *третьего окна прозрачности* оптического волокна со средней длиной волны 1550 нм. Это окно прозрачности выбрано как более «плоское», чем окно со средней длиной волны 1310 нм, то есть при той же ширине окна в 100 нм все волны этого диапазона затухают почти одинаково, что не наблюдается в окне со средней длиной волны 1310 нм.

Сети DWDM работают по принципу *коммутации каналов*, при этом каждая световая волна представляет собой отдельный *спектральный канал* и несет собственную информацию.

ПРИМЕЧАНИЕ

В технологии DWDM информация переносится модулированным синусоидальным сигналом. Известно, что такой сигнал может быть представлен набором синусоид, принадлежащих некоторому частотному спектру. Поэтому термин «волна» является достаточно условным, а более точным является термин «спектральный канал». Когда говорят «волна», то имеется в виду «центральная волна» диапазона спектра, отведенного для некоторого спектрального канала.

Манипуляции со световыми волнами в оборудовании DWDM основаны на физических явлениях, изученных классической оптикой и являющихся общими для электромагнитных колебаний любых видов. В главе 7 мы рассмотрели отрицательное влияние некоторых таких явлений на прохождение световых сигналов через оптическое волокно — например, искажение формы сигнала из-за хроматической или поляризационной дисперсии. В то же время оптические явления можно с пользой применить для операций мультиплексирования/демультиплексирования световых волн, их коммутации и усиления. Наиболее часто для манипуляций со световыми волнами используются следующие явления (рис. 9.1):

- **Дисперсия.** Она проявляется в том, что световые лучи *разной длины* имеют различные углы преломления на границе двух сред с различной оптической плотностью, что может быть использовано для мультиплексирования/демультиплексирования световых сигналов. Именно такой границей является грань призмы, поэтому красный луч проходит через призму под одном углом, синий — под другим и т. д.

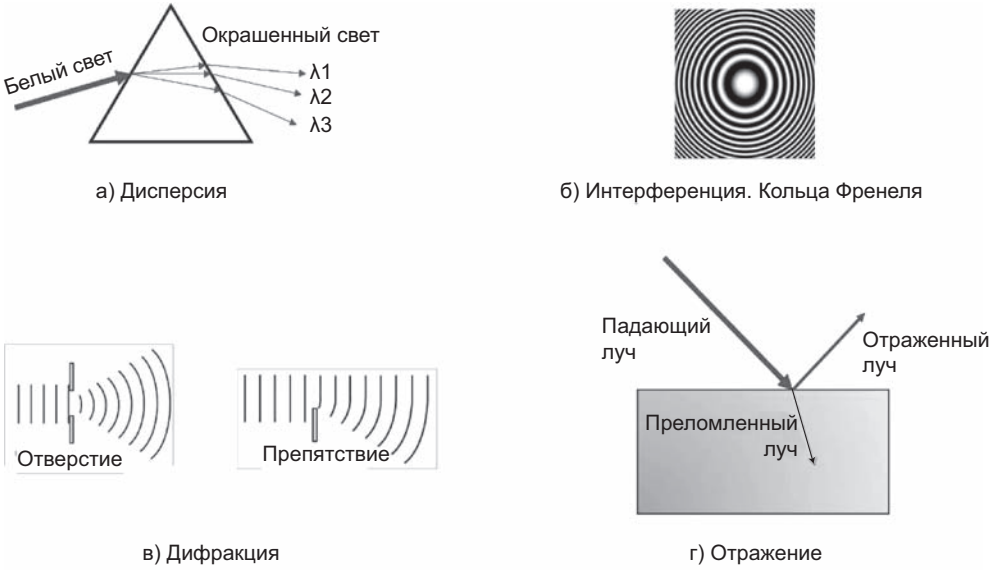


Рис. 9.1. Оптические эффекты

Иными словами, оптическая призма работает как *демультиплексор* неокрашенного светового сигнала. Ее можно использовать и в обратном направлении — как *мультиплексор* для смешения окрашенных лучей.

- **Интерференция.** Это эффект взаимодействия двух или более волн одной частоты, имеющих сдвиг по фазе. В зависимости от величины сдвига фазы интерференция может быть *конструктивной* — когда мощность результирующего сигнала увеличивается, или

деструктивной — когда мощность результирующего сигнала уменьшается. Максимум конструктивного эффекта достигается при нулевом сдвиге фазы между волнами — когда волны колеблются синфазно, а максимум деструктивного эффекта (то есть минимум интенсивности света) — при сдвиге фазы на половину периода, когда волны колеблются противофазно. Примером результата интерференции волн после прохождения круглого отверстия (диафрагмы) являются *кольца Френеля*. Помимо *усиления* или *ослабления* сигнала этот эффект, как мы увидим дальше, также можно использовать для демультиплексирования/мультиплексирования световых сигналов.

- **Дифракция.** Этот термин используется для описания различных явлений, но чаще всего — для описания эффектов отклонения от законов геометрической оптики при распространении лучей света. В таком терминологическом контексте дифракция означает, что распространение волны отклоняется от прямой линии при прохождении через отверстие или препятствие, размеры которого намного больше длины волны. Дифракция тесно связана с интерференцией. Так, огибание волной приводит к интерференции волн, если огибающие волны взаимодействуют с прямыми волнами, поскольку они отличаются по фазе. Дифракция может использоваться как механизм, вызывающий нужную интерференцию волн.
- **Отражение.** Полное отражение волны зеркалом может использоваться для ее направления в определенный волновод и тем самым ее *коммутацию* на определенный выходной порт устройства. В зависимости от угла падения и коэффициента преломления материала может наступить полное отражение световой волны. Частичное отражение полупрозрачным материалом может использоваться для интерференции прямой и отраженной волны с целью ее *усиления* или *подавления*. При дисперсии белого света часть энергии волн также отражается. В той части рис. 9.1, которая иллюстрирует эффект дисперсии, этот эффект не показан.

Технология DWDM является революционной не только потому, что она в десятки раз повысила верхний предел скорости передачи данных по оптическому волокну, но и потому, что открыла новую эру в технике мультиплексирования и коммутации, выполняя эти операции непосредственно над световыми сигналами, не прибегая к промежуточному преобразованию их в электрическую форму.

Во всех других технологиях, в которых световые сигналы также используются для передачи информации по оптическим волокнам, например SDH и высокоскоростной Ethernet, световые сигналы обязательно преобразуются в электрические, и только потом выполняется их мультиплексирование и коммутирование.

Заметим, что так как каждая световая волна представляет собой электромагнитное излучение с колебаниями определенной частоты (квантовую природу света мы не будем затрагивать), *мультиплексирование DWDM является частным случаем частотного мультиплексирования FDM*.

Основными функциями DWDM являются операции *мультиплексирования* и *демультиплексирования*, а именно — объединение различных волн в одном световом пучке и выделение информации каждого спектрального канала из общего сигнала. Поскольку оборудование DWDM почти всегда работает совместно с оборудованием SDH или OTN, оно выполняет для этих цифровых технологий функцию *кодирования* светового сигнала.

Частотные планы

Ширина спектра окна прозрачности, в котором работает оборудование DWDM, составляет примерно **10 000 ГГц**, или 100 нм. Разбиение этого диапазона на поддиапазоны, выделяемые для каждого отдельного спектрального канала, называется **частотным планом**.

Мультиплексирование DWDM называется «уплотненным» (dense) из-за того, что расстояние между длинами соседних волн в его частотном плане существенно меньше, чем в другой технологии, использующей тот же принцип мультиплексирования световых волн — технологии **грубого волнового мультиплексирования** (Coarse Wave Division Multiplexing, **CWDM**). В технологии CWDM ширина спектрального канала равна 20 нм, или 2556 ГГц. Из-за того что волны соседних спектральных каналов находятся на большом расстоянии друг от друга, сигнал отдельного канала легче выделять из общего светового сигнала и декодировать. Сегодня CWDM используется в различных скоростных вариантах технологии Ethernet, которые рассматриваются в главе 10.

До недавнего времени все частотные планы DWDM были фиксированными, то есть всем спектральным каналам отводились слоты частот равной ширины. Рекомендацией ITU-T G.694.1 для систем DWDM определено четыре *фиксированных частотных плана* с шагом (то есть расстоянием между центральными волнами каждого слота) в 100, 50, 25 и 12,5 ГГц. Эти планы показывают, что технология DWDM имеет право назваться уплотненной, так как ее шаг в любом варианте намного меньше шага в 2556 ГГц технологии CWDM.

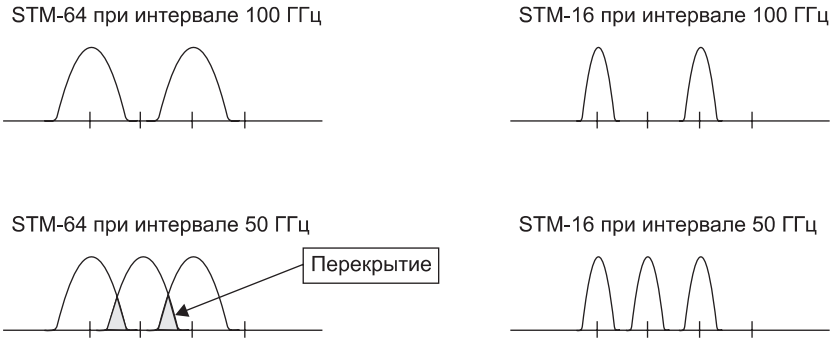


Рис. 9.2. Перекрытие спектра соседних волн для разных частотных планов и скоростей передачи данных

Оборудование DWDM первого поколения работало в основном с частотным планом с шагом 100 ГГц, или 0,8 нм. Использование плана с наиболее крупным шагом на начальном этапе развития технологии понятно, потому что реализация частотных планов с меньшим шагом (50, 25 и 12,5 ГГц) предъявляет гораздо более жесткие требования к оборудованию DWDM, особенно в том случае, если каждая волна переносит сигналы со скоростью выше 10 Гбит/с. Это легко объяснить, если вспомнить, что спектр сигнала тем шире, чем выше частота его модуляции (при фиксированной технике модуляции). Например, спектр сигнала STM-64 шире спектра сигнала STM-16 (рис. 9.2), что приводит к частичному перекрытию сигналов STM-64 у соседних волн при использовании плана 50 ГГц, в то время как при плане 100 ГГц такого перекрытия не происходит. Очевидно, что переход к более

узким слотам для увеличения числа волн требует применения более сложных методов кодирования, позволяющих уместить спектр сигнала в отведенную полосу пропускания.

В 2012 году организация ITU-T представила концепцию *гибкого частотного плана*, в соответствии с которой волновые каналы могут иметь различную ширину слота в зависимости от требуемой пропускной способности. Этот план определяет центр частотного слота с дискретностью 6,25 ГГц, при этом минимальная ширина слота должна быть кратна 12,5 ГГц. При гибком частотном плане допускается любая комбинация слотов различной ширины, при условии, что они не перекрываются. При реализации гибкого частотного плана в одном волокне могут сосуществовать каналы с различной шириной полосы пропускания и, соответственно, с различной пропускной способностью — например, 10 каналов с шириной полосы пропускания 100 ГГц и 60 каналов с шириной полосы пропускания 50 ГГц. Применение гибкого частотного плана значительно повышает эффективность магистрали DWDM, которая может быть ближе приспособлена к потребностям конкретной сети по сравнению с сетью, работающей с фиксированным частотным планом. Однако при этом оборудование становится более сложным.

Оборудование и топологии сетей DWDM

Сети DWDM поддерживают те же топологии, что и сети SDH, то есть линейную цепь, кольцо и произвольную смешанную топологию.

Для организации магистрали типа «линейная цепь» необходимо установить (рис. 9.3):

- **терминальные мультиплексоры DWDM** в ее конечных точках (на рисунке обозначены как T-MUX);
- **оптические мультиплексоры ввода-вывода** (Optical Add-Drop Multiplexer, **OADM**) в тех промежуточных точках, где имеется оборудование пользователей;
- **оптические усилители** (A), если между конечными точками требуется промежуточное усиление сигнала;
- **устройства компенсации дисперсии** (на рисунке не показаны), если между конечными точками требуется устранение дисперсии сигнала.

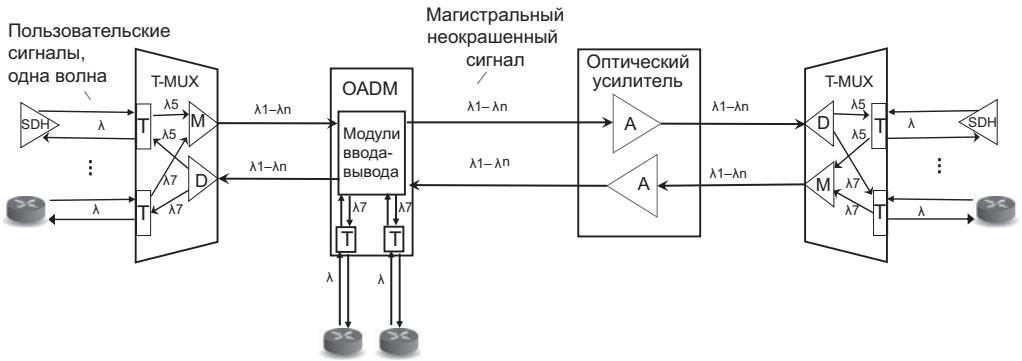


Рис. 9.3. Линейная цепь DWDM

В приведенной на рисунке схеме дуплексный обмен между абонентами сети (представленными оборудованием SDH или маршрутизаторами компьютерных сетей) происходит за счет однонаправленной передачи всего набора волн $\lambda 1$–λn по двум волокнам. Существует и другой вариант работы сети DWDM, когда для связи узлов сети используется одно волокно. Дуплексный режим достигается путем двунаправленной передачи оптических сигналов по одному волокну: половина волн частотного плана передает информацию в одном направлении, половина — в обратном.

Рассмотрим устройство и принципы работы типовых узлов линейной цепи DWDM. Заметим, что *кольцо DWDM* состоит из тех же элементов, что и линейная цепь, за исключением терминальных мультиплексоров. В кольце все мультиплексоры являются мультиплексорами ввода-вывода, поэтому отдельно организацию колец DWDM мы рассматривать не будем. Как и кольца SDH, кольца DWDM обладают повышенной отказоустойчивостью, присущей кольцевой топологии: всегда имеется возможность обойти отказавший участок кольца за счет направления сигнала в противоположную сторону.

Терминальные мультиплексоры

Терминальный мультиплексор T-MUX состоит из транспондеров (T) и блоков мультиплексирования (M) и демультиплексирования (D) (см. рис. 9.3).

Транспондер (transmitter-res**ponder)** является портом ввода-вывода пользовательских сигналов. Он преобразует оптический сигнал, поступающий от абонентского устройства пользователя (850 нм или 1310 нм), в сигнал из диапазона, в котором работает оборудование DWDM, и обратно. Например, транспондер может принимать сигнал SDH волны 1310 нм и преобразовывать его в волну 1531.1157 нм. Каждый транспондер поддерживает только одну определенную выходную волну диапазона DWDM (в примере это волна 1531.1157 нм). Такое фиксированное однозначное отображение *пользовательский сигнал — волна DWDM* может быть интерпретировано как приписывание абоненту некоторого идентификатора, который затем будет фигурировать в таблицах коммутации.

Для преобразования волн транспондеры сначала преобразуют принимаемый оптический сигнал в электрический, а затем электрический сигнал — в оптический сигнал новой длины волны, то есть работают по схеме O-E-O, подобно регенераторам и мультиплексорам SDH.

Для генерирования нужной волны диапазона DWDM транспондеры используют настраиваемые *лазеры*. Добавление новой волны в магистраль требует установки дополнительного транспондера. Транспондеры имеют мощные передатчики, обеспечивающие устойчивую работу без промежуточного усиления на расстояниях до 80 км.

Блок мультиплексирования (M) выполняет смешение окрашенных сигналов, получаемых от транспондеров, в магистральный неокрашенный сигнал. **Блок демультиплексирования** (D) выполняет обратную операцию. Обычно один и тот же блок может выполнять обе операции. Блок мультиплексор/демультиплексор DWDM может быть построен на основе разнообразных оптических механизмов.

В оптических мультиплексорах, поддерживающих сравнительно *небольшое количество длин волн* в волокне, применяются **тонкопленочные фильтры**. Тонкопленочный фильтр может представлять собой торец оптического волокна, скошенный под углом 30–45°, с нанесенными на него слоями покрытия, отличающимися толщиной и показателем преломления света. При определенной толщине и коэффициенте преломления слоев тонкопленочный фильтр *от-*

ражает (*фильтрует*) волну одной определенной длины, а остальные — пропускает. Таким образом, фильтр выделяет одну волну из общего неокрашенного сигнала. Для выделения всех волн тонкопленочные фильтры соединяют каскадом, что позволяет выполнить полное или частичное демультиплексирование неокрашенного сигнала. Этот же набор фильтров работает как мультиплексор, если окрашенные сигналы пропускаются в обратном направлении. Тонкопленочные фильтры вызывают весьма сильное затухание, поэтому при большом количестве составляющих фильтров в каскадном соединении сигнал становится слишком слабым, что и ограничивает количество волн в мультиплексоре такого типа до 16–32.

Для систем с *большим числом волн* требуются другие принципы фильтрации и мультиплексирования. Наиболее популярны так называемые интегральные **дифракционные фазовые решетки**, или **дифракционные структуры**, или **фазары**. Такой мультиплексор состоит из двух пластин, в которых свет распространяется свободно, рассеиваясь по всей области пластины (рис. 9.4). Пластины соединены набором волноводов (решеткой волноводов), по числу волн, которые нужно выделить из неокрашенного сигнала. На рисунке для упрощения показаны четыре таких волновода, то есть предполагается, что неокрашенный сигнал состоит из четырех волн. Приходящий неокрашенный сигнал λ_Σ попадает на входной порт. Затем этот сигнал свободно проходит через волновод-пластину и рассеивается по множеству волноводов. Сигнал в каждом из волноводов по-прежнему является мультиплексным, то есть каждая волна (λ_1, λ_2, λ_3, λ_4) остается представленной во всех волноводах.

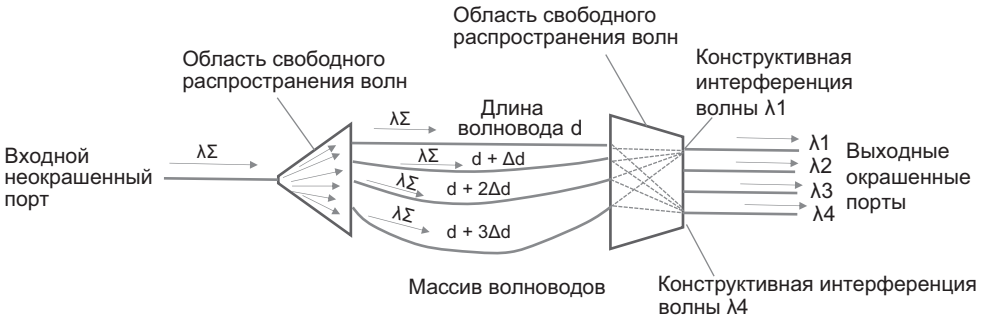


Рис. 9.4. Демультиплексирование сигнала с помощью дифракционной фазовой решетки

Важно, что все волноводы имеют *различную длину*, отличающуюся на постоянную величину шага Δd. После прохождения волноводов все неокрашенные волны попадают во вторую пластину, свободно распространяясь в ней. При этом сигналы, пришедшие из разных волноводов, имеют различную пространственную *фазу*. Этот сдвиг произошел из-за того, что каждый сигнал прошел путь различной длины по своему волноводу. В результате происходит интерференция волн и в пространстве второй пластины образуются максимумы (там, где произошла конструктивная интерференция) и минимумы (там, где произошла деструктивная интерференция) интенсивности волн. При этом области максимума и минимума для каждой из составляющих волн находятся в различных точках пространства пластины из-за дисперсии волн на границе пластины. Геометрия выходной пластины, соотношение коэффициентов преломления пластины и волноводов, а также шаг Δd выбираются так, чтобы область максимума некоторой волны находилась в районе ее выходного порта-волокна. На рисунке такие области показаны для двух волн: λ_1 и λ_4.

Мы рассмотрели процесс демультиплексирования неокрашенного света на его составляющие волны. Мультиплексирование происходит обратным путем, при этом отдельные окрашенные сигналы поступают на порты второй пластины, а из порта первой пластины выходит составной неокрашенный сигнал.

Дифракционные фазовые решетки выполняются в виде компактных интегральных модулей. Они являются ключевыми элементами мультиплексоров DWDM и обычно применяются для полного демультиплексирования светового сигнала, так как хорошо масштабируются и потенциально могут успешно работать в системах с сотнями спектральных каналов.

Оптические мультиплексоры ввода-вывода

Оптические мультиплексоры ввода-вывода (OADM) могут вывести (операция Drop) из неокрашенного сигнала волну определенной длины и ввести туда волну той же длины (операция Add), так что набор волн неокрашенного сигнала не изменится. При этом будет выполнено два соединения абонентов, подключенных к этому промежуточному мультиплексору.

Функциональная схема OADM представлена на рис. 9.5: мультиплексор включает два модуля ввода-вывода, каждый из которых выполняет операции ввода-вывода с одной волной λ_7. Левый модуль ввода-вывода работает с сигналами левого магистрального порта, а правый — правого. Операции ввода/вывода выполняются с абонентским оборудованием, подключенным к OADM через транспондеры.

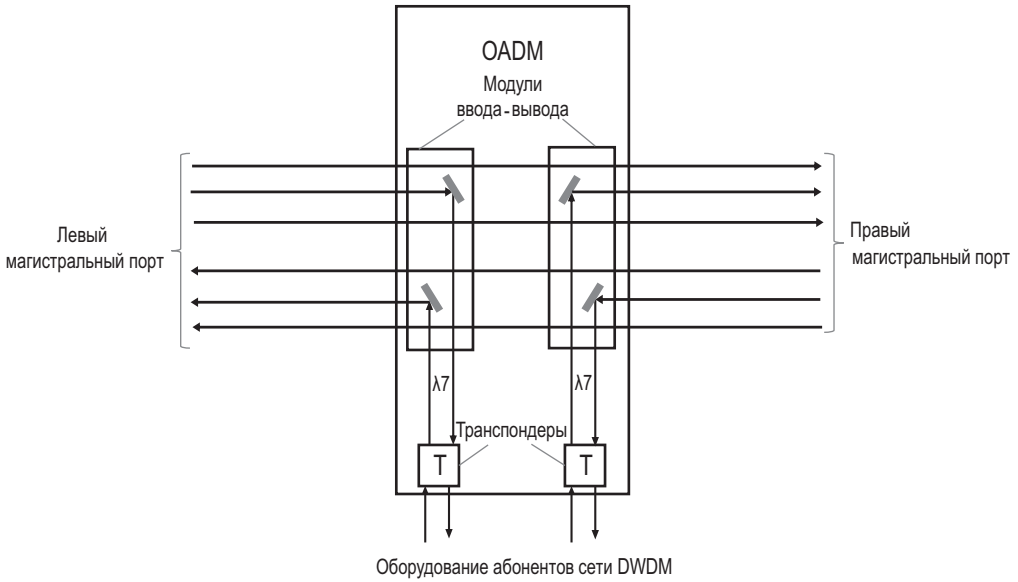


Рис. 9.5. Функциональная схема оптического мультиплексора ввода-вывода

Модули ввода-вывода строятся на основе *оптических фильтров* (тонкопленочных или другой конструкции). При выполнении операции вывода такой фильтр принимает неокрашен-

ный магистральный сигнал, отражает сигнал одной волны в выходной порт ввода-вывода, а сигналы всех остальных волн передает без изменения в виде выходного магистрального сигнала. При вводе волны фильтр работает в обратном направлении — он добавляет волну абонента (окрашенную волну) в общий магистральный сигнал. В соответствии с принципами работы сети DWDM во избежание коллизий фильтру не позволяется добавлять волну в магистральный сигнал, если она там уже существует. В примере, показанном на рис. 9.5, модуль ввода добавляет волну λ_7, потому что эта волна добавляется к сигналу уже после того, как она же была удалена из него другим модулем ввода-вывода, так что коллизии волн не наблюдается.

Модули ввода-вывода поддерживают операцию ввода-вывода только одной определенной волны — той, на которую настроен ее оптический фильтр. Для ввода-вывода нескольких волн нужно установить в OADM несколько блоков ввода-вывода, физически соединив их с соответствующими транспондерами.

Волоконно-оптические усилители

Практический успех технологии DWDM во многом определило появление **волоконно-оптических усилителей**. Эти оптические устройства усиливают световые сигналы в диапазоне 1550 нм, не прибегая к промежуточному преобразованию их в электрическую форму, как это делают регенераторы, применяемые в сетях SDH. Системы электрической регенерации сигналов весьма дороги и, кроме того, рассчитаны на определенный тип кодирования сигнала. Оптические усилители, прозрачно передающие информацию, позволяют наращивать скорость магистрали без необходимости модернизировать усилительные блоки из-за применения другого метода кодирования.

Наибольшее распространение в волоконно-оптических сетях получили усилители на *примесном волокне*, то есть волокне, легированном каким-либо редкоземельным элементом. Лазер усилителя, называемый **лазером накачки**, возбуждает атомы примесей в легированном волокне. При возвращении в нормальное состояние эти атомы излучают свет на той же длине волны и с той же фазой, что и внешний сигнал, требующий усиления. Примером такого усилителя является *усилитель EDFA* (Erbium Doped Fiber Amplifier), использующий примеси эрбия. Он имеет относительно узкую полосу усиления — 40 нм, поэтому применяется для усиления только части волн полного диапазона окна прозрачности 1550 нм.

Более совершенным типом усилителя для передачи данных на скоростях 100 Гбит/с и выше считается усилитель, использующий *эффект рассеяния света Рамана*[1]. При использовании рамановского усилителя энергия лазера накачки вызывает распределенное усиление сигнала в самом передающем волокне. Рамановский усилитель обладает более широкой полосой усиления, до 100 нм, что позволяет покрыть весь диапазон окна прозрачности 1550 нм, а значит, передавать большее количество волн в одном волокне. Кроме того, рамановский усилитель вносит меньше нелинейных шумов. Это позволяет увеличить максимальную длину участка между оптическими усилителями, которая может достигать 200 км и более.

---

[1] Чандрасекхара Раман открыл эффект рассеяния света в 1928 году, а в 1930 году получил за это Нобелевскую премию (в области физики).

Устройства компенсации дисперсии

Хроматическая дисперсия (см. главу 6) вносит основной вклад в искажение формы светового сигнала, что, в свою очередь, может приводить к ошибкам в распознавании передаваемых дискретных данных приемниками DWDM. Величина хроматической дисперсии зависит от длины волны, и, как следствие, сигналы различных волн, которыми оперирует технология DWDM, искажаются в разной степени, что делает трудной их компенсацию.

Для уменьшения эффекта хроматической дисперсии в сетях DWDM применяются *специальные волоконно-оптические кабели* со смещенной ненулевой дисперсией по стандарту G.655. Этот тип волокна не устраняет дисперсию полностью, но делает ее значительно меньшей, чем при использовании стандартного одномодового волокна G.652, и, что важно, приблизительно одинаковой для всех волн диапазона 1550 нм. Существуют также *устройства компенсации* хроматической дисперсии, которые устанавливаются в промежуточных узлах сети.

Компенсация другого вида дисперсии — *поляризационной модовой дисперсии* — требуется только при передаче данных на скоростях выше 10 Гбит/с. Используемое при этом оборудование значительно сложнее устройств компенсации хроматической дисперсии.

Применение цифровых сигнальных процессоров в приемниках оптических сигналов позволяет учесть эффекты дисперсии в алгоритмах выделения символов сигнала *программным способом*, поэтому в системах DWDM/OTN, где такие приемники используются, необходимость в установке устройств компенсации дисперсии отпадает.

Ячеистая топология и реконфигурируемые оптические кросс-коннекторы

По мере развития сетей DWDM в них все чаще стала применяться **ячеистая топология** (рис. 9.6), которая обеспечивает лучшие показатели в плане гибкости, производительности и отказоустойчивости, чем остальные топологии.

Для реализации ячеистой топологии необходим особый тип узлов, называемых **оптическими кросс-коннекторами** (Optical Cross-Connector, **OXC**), которые не только являются *мультиплексорами ввода-вывода*, то есть добавляют волны в общий транзитный сигнал и выводят их оттуда, но и поддерживают *произвольную коммутацию* между оптическими сигналами, передаваемыми волнами разной длины.

Возможности оптических кросс-коннекторов по созданию ячеистой топологии оцениваются количеством магистральных связей, которые они могут поддерживать со своими непосредственными соседями по сети. Эти связи проектировщики сетей DWDM называют **направлениями**. Так, верхний кросс-коннектор, изображенный на рис. 9.6, поддерживает четыре направления, а нижний — только два. (Нетрудно заметить, что мультиплексор ввода-вывода в линейной цепи или кольце всегда поддерживает только два направления.)

Существуют оптические кросс-коннекторы двух типов:

- **оптоэлектронные кросс-коннекторы** с промежуточным преобразованием сигналов в электрическую форму;
- **фотонные коммутаторы**, или **лямбда-маршрутизаторы**, — полностью оптические кросс-коннекторы.

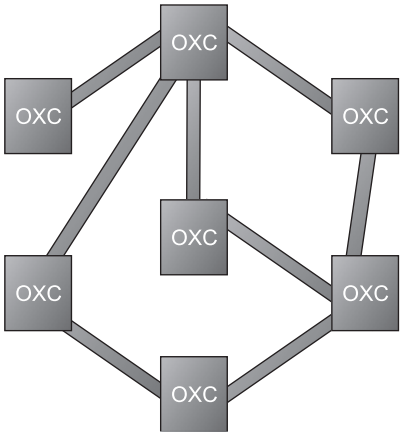


Рис. 9.6. Ячеистая топология сети DWDM

У оптоэлектронных кросс-коннекторов имеется принципиальное ограничение — они хорошо справляются со своими обязанностями при работе на скоростях до 2,5 Гбит/с, но на скоростях 10 Гбит/с и выше габариты таких устройств и потребление энергии превышают допустимые пределы. Фотонные коммутаторы свободны от такого ограничения.

Прочную позицию лидера фотонных коммутаторов занимают **реконфигурируемые оптические мультиплексоры ввода-вывода** (Reconfigurable Optical Add-Drop Multiplexors, **ROADM**). Эти устройства являются кросс-коннекторами, то есть позволяют выполнить коммутацию пользователей сети DWDM с ячеистой топологией, соединяя пользователей между собой без ограничений. Напомним, каждый пользователь представлен в сети волной определенной длины после преобразования его исходного сигнала транспондером.

Кроме того, ROADM являются *удаленно реконфигурируемыми* кросс-коннекторами. Под этим свойством понимается возможность администратора сети *программно* изменить конфигурацию таблицы коммутации этого устройства без необходимости физического добавления некоторых блоков, например транспондеров и модулей ввода-вывода. До появления ROADM для добавления новой волны инженер должен был лично явиться в точку присутствия оператора сети DWDM, чтобы установить новый модуль на шасси мультиплексора и сконфигурировать его. Поскольку ранние сети DWDM были достаточно статическими в отношении реконфигурации вводимых и выводимых потоков данных, то с необходимостью выполнять эту операцию путем физической перекоммутации операторы мирились. Развитие сетей DWDM привело к усложнению их топологии и повышению динамизма, когда появление новых клиентов сети стало достаточно частым явлением, а значит, операции добавления или выведения волн из магистрали стали выполняться регулярно и требовать более эффективной поддержки.

Один из возможных вариантов организации ROADM показан на рис. 9.7. Мультиплексор на этом рисунке поддерживает три магистральных направления (порты направлений 1, 2 и 3 связывают его с другими мультиплексорами), а также три *банка транспондеров* T, связанных через *клиентский кросс-коннектор* с портами ввода-вывода пользователей таким образом, что любой порт пользователя может быть соединен с любым транспондером из любого банка. Окрашенные сигналы в пределах одного банка транспондеров мультиплексируются в неокрашенный сигнал с помощью мультиплексора (на рисунке это «Бесцветный M/D») и передаются на *порт банка транспондеров*.

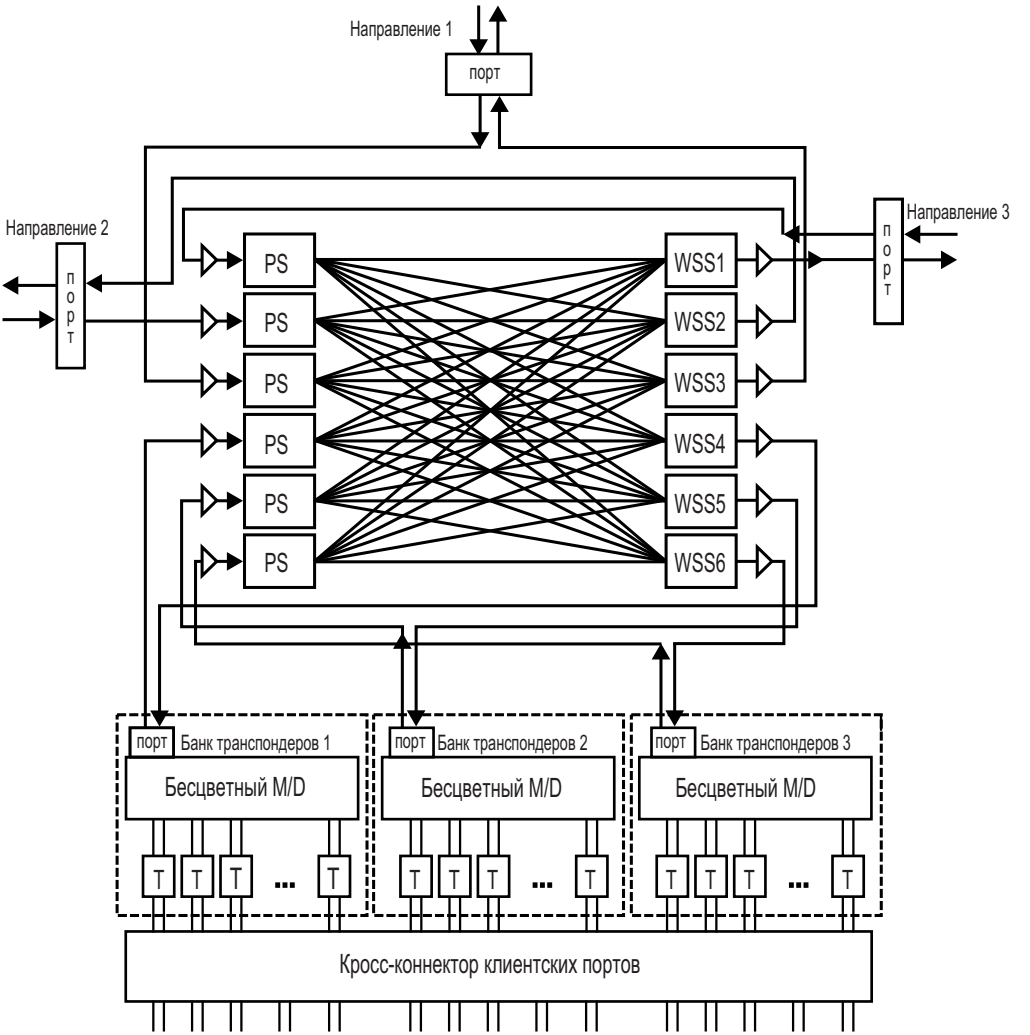


Рис. 9.7. Структурная схема ROADM

Задачей ROADM является маршрутизация волн, поступающих на все его порты, — от всех направлений и от всех банков транспондеров. Различают три возможности. В первом случае ROADM участвует в коммутации двух удаленных пользователей совместно с другими ROADM сети и выполняет транзит волны из одного направления в другое. Во втором случае выполняется коммутация внешнего пользователя с локальным пользователем, то есть коммутация порта направления с портом банка транспондеров. Третий вариант, когда коммутируются два локальных пользователя за счет направления волны из одного банка транспондеров в другой, теоретически возможен, но не имеет практического смысла.

Основой коммутатора ROADM являются **блоки селективной коммутации волн** (Wavelength Selective Switch, **WSS**), которые, собственно, и выполняют коммутацию. Каждый

из этих блоков имеет несколько входных портов и один выходной порт. На входные порты поступают *неокрашенные* сигналы как от каждого направления, так и от каждого банка транспондеров (в нашем примере каждый WSS имеет шесть входных портов, принимающих неокрашенные сигналы от трех направлений и от трех банков транспондеров). Для распределения каждого неокрашенного сигнала между шестью входными портами различных блоков WSS используются **оптические разветвители мощности** (Power Splitter, PS).

Функция блока WSS заключается в том, что он *выбирает* (select) в соответствии с таблицей коммутации из неокрашенного сигнала каждого своего входного порта по одной нужной волне, а затем мультиплексирует их и передает полученный неокрашенный сигнал в выходной порт. Выходной порт каждого из блоков WSS связан с портом одного из направлений или с портом одного из банков транспондеров.

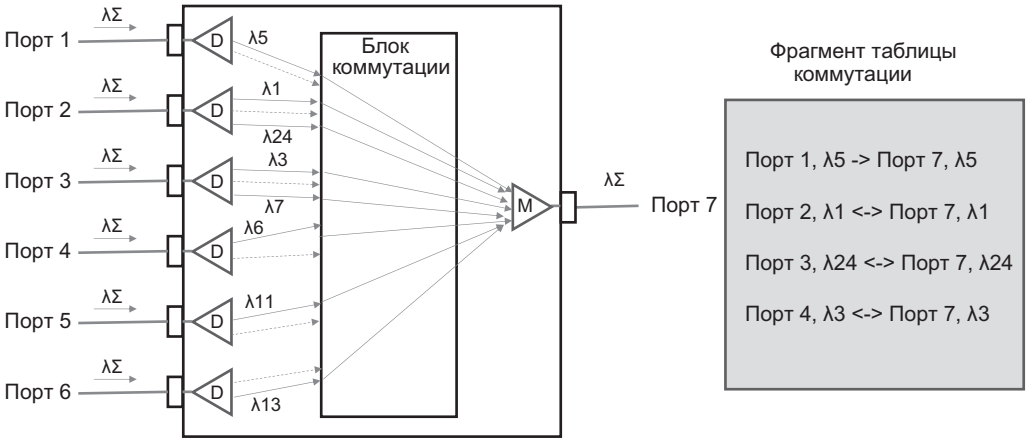


Рис. 9.8. Схема устройства WSS

На рис. 9.8 показана функциональная схема одного из возможных вариантов WSS, имеющего шесть входных портов 1–6 и один выходной порт 7. Блок WSS выполняет коммутацию следующим образом. После демультиплексирования блоками D (например, с помощью дифракционных решеток) окрашенные волны от каждого порта поступают по волноводам на *блок коммутации*. Блок коммутации на основании таблицы коммутации, фрагмент которой помещен на рисунке, выбирает из каждого полного набора волн (например, из 80, если ROADM поддерживает в волокне столько волн) только те, которые нужно передать на выходной порт, а остальные отбрасывает. Передаваемые волны показаны на рисунке сплошными линиями, а отброшенные — пунктирными (для ясности на рисунке показано только несколько волн). Естественно, в выходной порт не могут попасть две одинаковые волны из разных входных портов, поскольку это запрещенная комбинация.

Чтобы RODAM мог выполнять коммутацию своих локальных пользователей с любой волной любого направления, количество банков транспондеров должно быть равно количеству *N* направлений, поддерживаемых ROADM. Действительно, так как одна и та же волна присутствует в одном экземпляре в каждом из *N* направлений ROADM, она может быть использована для соединения *N* различных пользователей, подключенных через кросс-коннектор клиентских портов к данному ROADM, с внешними пользователями.

Без соответствия количества банков транспондеров количеству направлений удаленное реконфигурирование не всегда будет возможно — например, если необходимо ввести-вывести одну и ту же волну из всех направлений, а количество банков транспондеров меньше, чем количество направлений. В обычном, нереконфигурируемом OADM, такая проблема решается за счет физической установки нового модуля транспондеров, *когда в этом возникает необходимость*. Для удаленного реконфигурирования все необходимые ресурсы мультиплексора должны быть *всегда* в наличии, чтобы оставалось только удаленно изменить связи между ними.

Блок коммутации волн может быть реализован разными средствами. Один из наиболее часто применяемых вариантов — **микроэлектронная механическая система**, использующая отражение света и представляющая собой набор подвижных зеркал очень маленького (диаметром менее миллиметра) размера (рис. 9.9). Микроэлектронная механическая система получает составляющие волны от демультиплексора по различным волноводам. За счет поворота микрозеркала на заданный угол исходный луч определенной волны направляется в соответствующий выходной волновод.

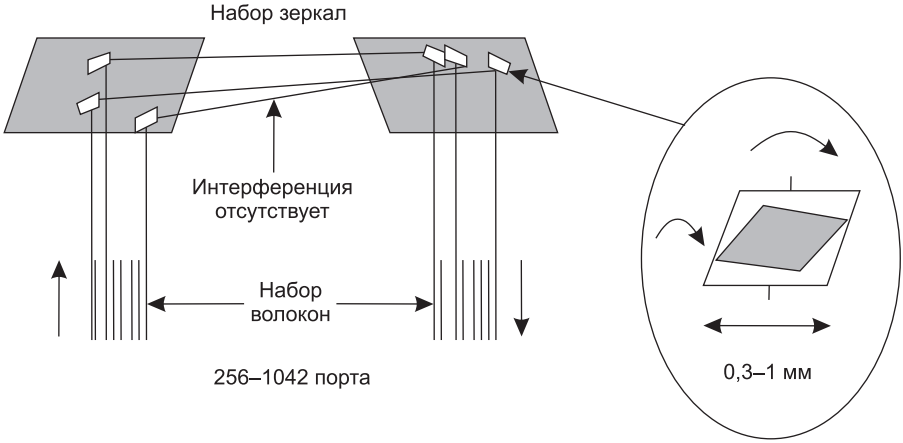


Рис. 9.9. Микроэлектронная механическая система кросс-коммутации

Особенностью представленной архитектуры ROADM является возможность направить любую волну из любого направления в любой пользовательский порт ввода-вывода. Говорят, что ROADM, обладающий такой функциональностью, является **неокрашенным** и **ненаправленным**. Необходимо подчеркнуть, что не каждое устройство ROADM является неокрашенным ненаправленным, так как название устройства этого типа отражает только возможность его программного реконфигурирования, но не степень его гибкости. Например, устройства ROADM первого поколения имели только один банк транспондеров, так что пользовательские порты ввода-вывода были физически привязаны к определенной волне и программное реконфигурирование позволяло вывести определенную волну только в определенный порт, а изменение волны для некоторого пользователя требовало физического переключения его кабеля к другому порту. Поэтому такие устройства ROADM не могли быть названы неокрашенными. Аналогично обстояло дело и с ненаправленностью, так как эти устройства ROADM были рассчитаны на кольцевые или линейные топологии и поддерживали только два направления.

Вопросы к части II

1. Может ли линия связи быть составным каналом?
2. Укажите, к какому типу линий связи относятся волоконно-оптические кабели:
   а) ненаправленные проводные;
   б) направленные проводные;
   в) направленные кабельные.
3. Чем отличаются усилители и регенераторы телекоммуникационных сетей?
4. Сетевой адаптер является устройством:
   а) DCE;
   б) DTE;
   в) DCE и DTE одновременно.
5. Какова частота электромагнитных колебаний волны длиной 1550 нм?
6. Можно ли выполнить преобразование Фурье для непериодической функции?
7. Какие меры можно предпринять для увеличения информационной скорости звена (выберите вариант ответа):
   а) уменьшить длину кабеля;
   б) выбрать кабель с меньшим сопротивлением;
   в) выбрать кабель с более широкой полосой пропускания;
   г) применить метод кодирования с более узким спектром.
8. Поясните, почему полезно знать амплитудно-частотную характеристику линии (выберите вариант ответа):
   а) она позволяет узнать затухание сигнала определенной амплитуды и любой частоты;
   б) она позволяет узнать искажения сигнала определенной формы.
9. Дайте определение порога чувствительности приемника.
10. Проверьте, достаточна ли для устойчивой передачи данных мощность передатчика в 40 дБм, если длина кабеля равна 70 км, погонное затухание кабеля — 0,5 дБ/км, а порог чувствительности приемника составляет 20 дБм.
11. Что является причиной перекрестных наводок на ближнем конце кабеля?
12. Какой кабель более качественно передает сигналы — с большим значением параметра NEXT или с меньшим?
13. Какой тип кабеля предназначен для передачи данных на большие расстояния (выберите вариант ответа):
    а) многомодовый;
    б) одномодовый;
    в) витая пара.

14. Каким будет теоретический предел скорости передачи данных в битах в секунду по линии связи с шириной полосы пропускания 100 мГц, если мощность передатчика составляет 10 дБм, а мощность шума в линии связи равна 2 дБм?
15. Можно ли кодировать дискретную информацию аналоговыми сигналами?
16. Какой будет скорость передачи данных на линии, если частота изменения сигнала равна 100 бод, а сигнал имеет 16 состояний?
17. Несут ли информацию импульсы на тактовой шине компьютера?
18. Регенерацией оптического сигнала называется (выберите вариант ответа):
    а) усиление сигнала одним из типов оптических усилителей;
    б) восстановление формы сигнала за счет преобразования в электрическую форму;
    в) восстановление формы сигнала за счет преобразования в электрическую форму и обратно.
19. Сколько бит за один такт передается при кодировании 64-QAM?
20. Поясните, из каких соображений выбрана частота дискретизации 8 кГц в методе квантования PCM.
21. Спектр какого сигнала у́же при одной и той же тактовой частоте передатчика (выберите вариант ответа):
    а) потенциального кода;
    б) амплитудной модуляции.
22. Чем логическое кодирование отличается от физического?
23. Укажите, каким образом можно повысить скорость передачи данных по кабельной линии связи:
    а) уменьшить спектр сигнала за счет применения более совершенного метода кодирования/модуляции и повысить тактовую частоту сигнала;
    б) увеличить число символов кода и сохранить ту же тактовую частоту сигнала.
24. Укажите, какими способами можно улучшить свойство самосинхронизации кода NRZI:
    а) использовать логическое кодирование, исключающее появление длинных последовательностей единиц;
    б) скремблировать данные;
    в) использовать логическое кодирование, исключающее появление длинных последовательностей нулей.
25. Какой режим временного мультиплексирования используется в сетях с коммутацией пакетов?
26. Найдите первые две гармоники спектра NRZ-сигнала при передаче последовательности 110011001100..., если тактовая частота передатчика равна 100 МГц.
27. Во сколько раз увеличится ширина спектра кода NRZ при увеличении тактовой частоты передатчика в 3 раза?
28. Укажите, какими из следующих характеристик обладают первичные сети:
    а) поддерживают технику коммутации каналов;
    б) мультиплексируют данные на основании техники асинхронного TDM;
    в) предоставляют услуги канала «точка-точка».

29. Цифровой кросс-коннектор первичной сети отличается от мультиплексора ввода-вывода тем, что он (выберите вариант ответа):
    а) имеет более чем два магистральных порта;
    б) коммутирует данные в цифровом формате;
    в) может выполнять коммутацию любых подканалов магистральных каналов с каналами пользователей.
30. Если при планировании первичной сети количество пользователей и интенсивности генерируемых ими потоков данных были оценены неверно, то (выберите вариант ответа):
    а) это приведет к длительным задержкам передачи пользовательских данных;
    б) к невозможности обслуживания некоторого количества пользователей.
31. Кадрирование информации в первичной сети преследует цель (выберите вариант ответа):
    а) локализации и исправления ошибок;
    б) мониторинга качества работы сети;
    в) передачи пульсирующего трафика.
32. Мультикадры в первичных сетях используются для (выберите вариант ответа):
    а) передачи длинных полей данных компьютерных пакетов;
    б) распределения длинных полей заголовка кадра между несколькими кадрами;
    в) компрессии заголовка кадра.
33. Укажите, какие варианты организации услуг первичной сети являются реализуемыми:
    а) PDH/DWDM;
    б) OTN/DWDM;
    в) SDH/PDH.
34. Почему кольцо является одной из самых популярных топологий сети SDH?
35. Технология DWDM повысила скорости магистралей первичных сетей за счет (выберите вариант ответа):
    а) использования других окон прозрачности;
    б) применения новых схем кодирования;
    в) параллельной передачи данных по нескольким волнам.
36. Для выделения отдельной волны из неокрашенного сигнала можно использовать следующие световые эффекты:
    а) дифракцию;
    б) дисперсию;
    в) интерференцию.
37. Назовите преимущества гибкого частотного плана DWDM.
38. Транспондер преобразует длину волны, используя (выберите вариант ответа):
    а) эффект интерференции;
    б) эффект дисперсии;
    в) преобразование O-E-O.

39. Почему волноводы дифракционной решетки имеют разную длину?
40. Может ли мультиплексор ввода-вывода ввести в магистраль волну длины λ1, если он ее вывел?
41. Главной особенностью устройств ROADM является (выберите вариант ответа):
    а) наличие блоков WSS;
    б) поддержка удаленной программной реконфигурации.
42. Технология OTN унаследовала от технологии SDH (выберите вариант ответа):
    а) иерархию скоростей с коэффициентом 4;
    б) технику виртуальных контейнеров;
    в) побайтное мультиплексирование.
43. Приведите аргументы за и против двух подходов к перевозке пассажиров в примере-аналогии.
44. Каким образом кадры OTN размером в 15 232 байта переносят кадры STM-16 размером в 38 880 байт?
45. Трибутарный слот OTN имеет (выберите вариант ответа):
    а) фиксированную скорость;
    б) фиксированный объем;
    в) переменный объем.
46. Макспондер OTN выполняет функции (выберите вариант ответа):
    а) ROADM;
    б) терминального мультиплексора;
    в) транспондера.

Часть III

Технология Ethernet

- Глава 10. Ethernet в локальных сетях
- Глава 11. Отказоустойчивые и виртуальные локальные сети
- Глава 12. Ethernet операторского класса

Технология Ethernet является сегодня монополистом — сначала она вытеснила все остальные технологии канального уровня из локальных сетей, а затем сделала то же самое и в области глобальных сетей. Трудно сказать, почему именно это произошло — то ли по причине простоты технологии, а значит, и низкой стоимости оборудования и его эксплуатации, то ли из-за удачного названия, то ли вследствие необыкновенного везения, полагает изобретатель Ethernet Роберт Меткалф (Robert Metcalfe), состоявшего в том, что «каждый раз, когда появлялось что-то на замену Ethernet, люди, ответственные за продвижение новой технологии, снова выбирали для нее название Ethernet». Но факт остается фактом — как в локальных, так и в глобальных проводных сетях под уровнем IP работает технология Ethernet.

Технология Ethernet прошла большой путь. Долгое время, примерно до середины 2000-х годов, она применялась исключительно в локальных сетях. Практически во всех технологиях 80-х, включая Ethernet, использовалась *разделяемая среда* — как удобное и экономичное средство объединения компьютеров на физическом уровне. С середины 90-х в локальных сетях стали применяться *коммутируемые* версии технологий. Отказ от разделяемой среды позволил повысить производительность (на сегодня — до 800 Гбит/с, но похоже, что это не предел) и масштабируемость локальных сетей. Преимуществом коммутируемых локальных сетей является также возможность логической структуризации сети с разделением ее на отдельные сегменты, называемые виртуальными локальными сетями (VLAN).

Успех Ethernet в локальных сетях привел к тому, что стало целесообразным ее применение и в глобальных сетях для создания всеохватных бесшовных сервисов Ethernet, работающих по принципу «из конца в конец». После придания Ethernet некоторых дополнительных функций появилась версия Ethernet операторского класса (Carrier Grade Ethernet), работающая ныне на канальном уровне в подавляющем большинстве глобальных связей.

В главе 10 рассматриваются все версии протокола Ethernet — от первой версии 10G до последних версий 100G, 200G, 400G и 800G. Здесь также изучается алгоритм моста, лежащий в основе современных коммутаторов локальных сетей, и основные принципы работы таких коммутаторов.

В главе 11 изучаются вопросы обеспечения отказоустойчивости локальных коммутируемых сетей с помощью протокола покрывающего дерева (STP), а также техника виртуальных локальных сетей (VLAN), позволяющая быстро и эффективно выполнять логическую структуризацию сети.

Глава 12 посвящена Ethernet операторского класса. В ней рассматриваются усовершенствования, внесенные в технологию для более успешного применения Ethernet в сетях операторов связи.

ГЛАВА 10 Ethernet в локальных сетях

Первый этап — разделяемая среда

Сегодня технологии со случайным доступом к разделяемой среде применяются только в беспроводных сетях (например, в сетях Wi-Fi, см. главу 25). В проводных локальных сетях уже довольно давно, с середины 90-х годов, разделяемая среда не используется из-за плохой масштабируемости такого подхода. И хотя в стандартах единственной выжившей технологии локальных проводных сетей — Ethernet — вариант работы на разделяемой среде все еще описан, он разрешен к применению только для низко- и среднескоростных версий Ethernet, но не для скоростей 10 Гбит/с и выше.

Тем не менее мы включили в книгу описание основных идей и характеристик Ethernet и других технологий на разделяемой проводной среде, так как это помогает понять особенности техники применения разделяемой среды, что полезно при разработке новых технологий беспроводных сетей, где эта техника является естественной. Кроме того, знание истории развития Ethernet помогает лучше понять некоторые ее характеристики, такие, например, как размер и формат кадра Ethernet, сохранившиеся и в современных коммутируемых версиях Ethernet, поскольку они были выбраны с учетом их надежной работы на разделяемой среде.

Стандартная топология и разделяемая среда

Основная цель, которую ставили перед собой разработчики первых локальных сетей во второй половине 70-х годов, заключалась в нахождении простого и дешевого решения для объединения в вычислительную сеть нескольких десятков компьютеров, находящихся в пределах одного здания.

Для упрощения и, соответственно, удешевления аппаратных и программных решений разработчики первых локальных сетей остановились на совместном использовании **общей разделяемой среды передачи данных**. Разделяемость среды означает, что все узлы сети пользуются ею, но по очереди, и в любой момент времени по ней передается кадр только одного узла.

Уточним, что этот метод связи компьютеров впервые был опробован при создании радиосети ALOHA Гавайского университета в начале 70-х под руководством Нормана Абрамсона. Радиоканал определенного диапазона частот естественным образом является общей средой для всех передатчиков, использующих частоты этого диапазона для кодирования данных. Сеть ALOHA работала по методу случайного доступа, когда каждый узел мог начать передачу пакета в любой момент времени. Если после этого узел не дожидался подтверждения приема в течение определенного тайм-аута, то он посылал этот пакет снова. Общим был

радиоканал с несущей частотой 400 МГц и полосой 40 кГц, что обеспечивало передачу данных со скоростью 9600 бит/с.

Немного позже Роберт Меткалф, работая в компании Xerox, повторно реализовал идею разделяемой среды уже для проводного варианта технологии LAN. Непрерывный сегмент коаксиального кабеля стал аналогом общей радиосреды. Все компьютеры присоединялись к этому сегменту кабеля по схеме монтажного ИЛИ, поэтому при передаче сигналов одним из передатчиков все приемники получали один и тот же сигнал, как и при использовании радиоволн. На рис. 10.1 — начало служебной записки Роберта Меткалфа от 22 мая 1973 года, с наброском разделяемой среды на коаксиальном кабеле, где эта среда названа «a cable-tree ether», что можно приблизительно перевести как «древовидный кабельный эфир».

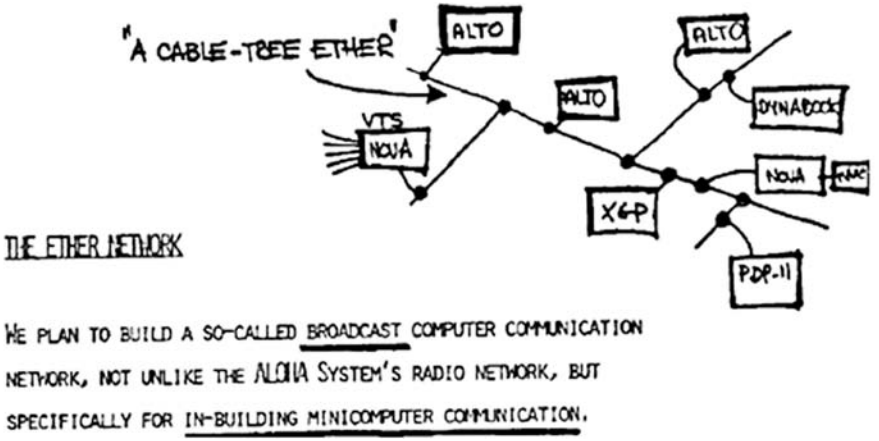


Рис. 10.1. Рисунок Роберта Меткалфа с иллюстрацией идеи эмуляции разделяемого радиоэфира при помощи коаксиального кабеля (https://ethernethistory.typepad.com/papers/ethernetbobmemo.pdf)

В технологиях Token Ring и FDDI тот факт, что компьютеры используют разделяемую среду, не так очевиден, как в случае Ethernet. Физическая топология (см. ранее) этих сетей — кольцо: каждый узел соединяется кабелем с двумя соседними узлами (рис. 10.2). Однако и эти отрезки кабеля являются разделяемыми, так как в каждый момент времени только один компьютер может использовать кольцо для передачи своих пакетов, а именно — тот, у которого в данный момент находится специальный кадр, называемый токеном или маркером.

> Простые **стандартные топологии физических связей** (звезда у коаксиального кабеля Ethernet и кольцо у Token Ring и FDDI) обеспечивают простоту разделения кабельной среды.

Использование разделяемых сред позволяет ***упростить*** логику работы узлов сети. Действительно, поскольку в каждый момент времени выполняется только одна передача, отпадает необходимость в буферизации кадров в транзитных узлах и, как следствие, в самих транзитных узлах. Соответственно, отпадает и необходимость в сложных процедурах управления потоком как борьбы с перегрузками.

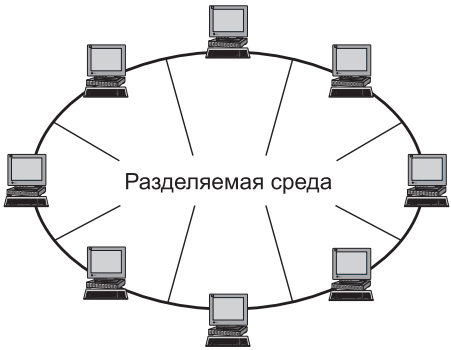


Рис. 10.2. Разделяемая среда в кольцевых топологиях

Основной недостаток разделяемой среды — плохая масштабируемость. Этот недостаток является принципиальным, поскольку, независимо от метода доступа к среде, ее пропускная способность делится между всеми узлами сети. Здесь применимо положение теории очередей (см. главу 5): как только коэффициент использования общей среды превышает определенный порог, очереди к среде начинают расти нелинейно, и сеть становится практически неработоспособной. Значение порога зависит от метода доступа. Так, в сетях ALOHA это значение является крайне низким — всего около 18 %, в сетях Ethernet — около 30 %, а в сетях Token Ring и FDDI оно возросло до 60–70 %.

Локальные сети, являясь пакетными сетями, используют принцип TDM, но, в отличие от первичных сетей, это *асинхронная* версия TDM.

Алгоритм *управления доступом к среде* является одной из важнейших характеристик любой технологии LAN на разделяемой среде, в значительно большей степени определяя ее облик, чем метод кодирования сигналов или формат кадра. В технологии Ethernet в качестве алгоритма разделения среды применяется *метод случайного доступа.* И хотя его трудно назвать совершенным — при росте нагрузки полезная пропускная способность сети резко падает, — он, благодаря своей простоте, стал основой успеха технологии Ethernet в 1980-е годы. Технологии Token Ring и FDDI используют *метод маркерного доступа*, основанный на передаче от узла к узлу особого кадра — маркера (токена) доступа. При этом только узел, владеющий маркером доступа, имеет право доступа к разделяемому кольцу. Более детерминированный характер доступа технологий Token Ring и FDDI предопределил более эффективное использование разделяемой среды, чем у технологии Ethernet, но одновременно и усложнил оборудование. Чтобы описать особенности алгоритма доступа к среде технологии Ethernet, нужно сначала познакомиться с некоторыми базовыми характеристиками этой технологии — такими, как разбиение ее функций на уровни, способ адресации и формат кадра.

Уровни Ethernet

Изначально в фирменной версии Ethernet, принятой компаниями DEC, Intel и Xerox (и получившей название DIX Ethernet), ее функции не разделялись на уровни, то есть это была монолитная технология. Такими же, к слову, были и другие технологии локальных сетей 70-х: Token Ring и Arcnet. При стандартизации этих технологий в институте IEEE

в 1980 году был организован комитет 802, результатом работы которого стало принятие семейства стандартов IEEE 802.x, которые обобщили фирменный опыт создателей протоколов локальных сетей и выделили в них три функциональных уровня (рис. 10.3):

- физический уровень;
- уровень доступа к среде (Media Access Control, MAC);
- уровень управления логическим каналом (Logical Link Control, LLC).

Определение физического уровня IEEE 802 полностью соответствует определению физического уровня модели OSI. В нем описаны характеристики технологии: возможные типы физической среды, методы кодирования, типы соединительных разъемов и т. п. А вот функции канального уровня модели OSI в стандартах IEEE 802 представлены двумя уровнями — MAC и LLC. Выделение функций доступа к среде в отдельный уровень отражает их важность при использовании разделяемой среды.

Стандарты уровня MAC описывают различные методы доступа к разделяемой среде, используемые в технологиях локальных сетей того времени. Для стандартизации каждого метода были созданы отдельные рабочие группы комитета 802. Рабочая группа 802.3 занялась стандартизацией метода доступа технологии Ethernet. На рис. 10.3 для полноты картины показаны также некоторые другие рабочие группы, как прекратившие свое существование (группа 802.5, занимавшаяся стандартизацией технологии Token Ring), так и действующие поныне (группа 802.11, занимающаяся стандартизацией беспроводной технологии Wi-Fi).

Уровень MAC выполняет две основные функции: предоставление узлу доступа к среде и последующая доставка кадра от узла источника к узлу назначения в соответствии с его адресом. Уровень MAC дал название адресам локальных сетей всех технологий локальных сетей, в том числе адресам Ethernet, так что оба названия (**MAC-адрес** и **адрес Ethernet**) используются как равноправные. Хотя уровень MAC и отвечает за доставку кадра узлу назначения, он не обеспечивает его надежную доставку. Эту функцию реализует уровень LLC, назначение которого сравнимо с назначением транспортного уровня модели OSI или протокола TCP в стеке TCP/IP.

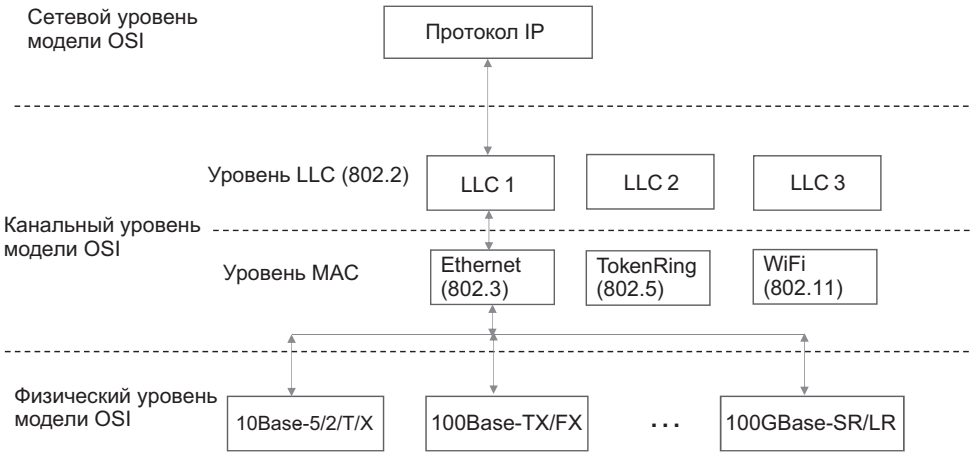


Рис. 10.3. Уровни стандартов IEEE 802.x

Специалисты рабочей группы 802.2, разработавшие протоколы уровня LLC, хотели обеспечить законченность технологиям локальных сетей, выражающуюся в способности самостоятельно, без помощи сетевого уровня обеспечивать транспортные услуги любого типа. На уровне LLC было определено три типа услуг:

- **Услуга LLC1** — услуга *без установления соединения и без подтверждения получения данных*.
- **Услуга LLC2** — услуга, позволяющая пользователю установить *логическое соединение* перед началом передачи любого блока данных и, если это требуется, выполнить *процедуры восстановления* после ошибок и упорядочивание потока блоков в рамках установленного соединения.
- **Услуга LLC3** — услуга *без установления соединения, но с подтверждением получения данных*.

Какой из трех режимов работы уровня LLC будет использован, зависит от требований протокола верхнего уровня. Нужно сказать, что на практике идея обобщения функций обеспечения надежной передачи кадров в общем уровне LLC не оправдала себя. Технология Ethernet в версии DIX изначально функционировала в наиболее простом дейтаграммном режиме — в результате оборудование Ethernet и после опубликования стандарта IEEE 802.2 продолжало поддерживать только этот режим работы, который формально является режимом LLC1.

MAC-адреса

На уровне MAC используются регламентированные стандартом IEEE 802.3 уникальные 6-байтовые адреса. Обычно MAC-адрес записывают в виде шести пар шестнадцатеричных цифр, разделенных дефисами или двоеточиями, например 11-A0-17-3D-BC-01. Каждый сетевой адаптер имеет по крайней мере один MAC-адрес.

Помимо отдельных интерфейсов, MAC-адрес может определять группу интерфейсов и даже все интерфейсы сети. Первый (младший) бит старшего байта адреса назначения — это признак того, является адрес индивидуальным или групповым. Если он равен 0, то адрес является **индивидуальным**, то есть идентифицирует один сетевой интерфейс, а если 1, то **групповым**. Групповой адрес связан только с интерфейсами, сконфигурированными (вручную или автоматически по запросу вышележащего уровня) как члены группы, номер которой указан в групповом адресе. Если сетевой интерфейс включен в группу, то, наряду с уникальным MAC-адресом, с ним ассоциируется еще один адрес — групповой. В частном случае, если групповой адрес состоит из всех единиц (имеет шестнадцатеричное представление 0xFFFFFFFFFFFF), то он идентифицирует все узлы сети и называется **широковещательным**.

Второй бит старшего байта адреса определяет способ назначения адреса — **централизованный** или **локальный**. Если этот бит равен 0 (что бывает почти всегда в стандартной аппаратуре Ethernet), то это говорит о том, что адрес назначен централизованно по правилам IEEE 802.

Комитет IEEE распределяет между производителями оборудования так называемые **организационно уникальные идентификаторы** (Organizationally Unique Identifier, OUI). Каждый производитель помещает выделенный ему идентификатор в три старших байта адреса (например, идентификатор 0x0020AF определяет компанию 3COM, а 0x00000C — Cisco).

За уникальность младших трех байтов адреса отвечает производитель оборудования. 24 бита, отводимые производителю для адресации интерфейсов его продукции, позволяют выпустить примерно 16 миллионов интерфейсов под одним идентификатором организации. Уникальность централизованно распределяемых адресов распространяется на все основные технологии локальных сетей — Ethernet, Token Ring, FDDI и т. д. Локальные адреса назначаются администратором сети, в обязанности которого входит обеспечение их уникальности.

Сетевые адаптеры Ethernet могут работать и в так называемом **«неразборчивом» режиме** (promiscuous mode), захватывая все кадры, поступающие на интерфейс, независимо от их MAC-адресов назначения. Обычно такой режим используется для мониторинга трафика, когда захваченные кадры изучаются затем для нахождения причины некорректного поведения некоторого узла или отладки нового протокола.

В конце 1990-х годов был принят новый стандарт MAC-адресов EUI-64, расширяющий формат MAC-адреса до 8 байт, то есть 64 бит вместо 48. Это было сделано для увеличения адресного пространства узлов технологий канального уровня, использующих MAC-адреса, — Ethernet, Wi-Fi и некоторых других. Оба формата — 48 бит и 64 бита — используются в настоящее время сетевым оборудованием, 48-битный формат при использовании протокола IPv4, а 64-битный — при использовании протокола IPv6. Это связано с тем, что 8 байт MAC-адреса не создают проблему при автоконфигурировании адреса IPv6 путем использования MAC-адреса узла в качестве части IPv6-адреса, а в короткий 4-байтный адрес IPv4 его включить нельзя. В дальнейшем изложении мы будем подразумевать по умолчанию использование 48-битных MAC-адресов, а в случае использования 64-битных будем указывать на это явно.

Форматы кадров технологии Ethernet

Существует несколько стандартов формата кадра Ethernet. На практике в оборудовании Ethernet используется только один формат кадра, а именно — кадр Ethernet DIX, называемый иногда кадром Ethernet II по номеру последнего стандарта DIX (рис. 10.4).

6 байт	6 байт	2 байта	46–1500 байт	4 байта
DA	SA	T	Данные	FCS

Рис. 10.4. Формат кадра Ethernet DIX (II)

Первые два поля заголовка отведены под адреса:

- **DA (Destination Address)** — MAC-адрес узла назначения.
- **SA (Source Address)** — MAC-адрес узла отправителя.
- Для доставки кадра достаточно одного адреса — адреса назначения; адрес источника помещается в кадр для того, чтобы узел, получивший кадр, знал, от кого пришел кадр и кому нужно на него ответить. Принятие решения об ответе не входит в компетенцию протокола Ethernet — это дело протоколов верхних уровней, а Ethernet выполнит такое действие, если с сетевого уровня поступит соответствующее указание.

- **Поле T (Type, EtherType)** содержит условный код протокола верхнего уровня, данные которого находятся в поле данных кадра, например, шестнадцатеричное значение 08-00 соответствует протоколу IP. Это поле требуется для поддержки интерфейсных функций мультиплексирования и демультиплексирования кадров при взаимодействии с протоколами верхних уровней.
- **Поле данных** может содержать от 46 до 1500 байт. Если длина пользовательских данных меньше 46 байт, то это поле дополняется до минимального размера байтами заполнения. Эта операция требуется для корректной работы метода доступа Ethernet (см. следующий раздел).
- **Поле контрольной последовательности кадра** (Frame Check Sequence, FCS) состоит из 4 байт контрольной суммы. Это значение вычисляется по алгоритму CRC-32.

Кадр Ethernet DIX (II) не отражает разделения канального уровня Ethernet на уровни MAC и LLC: его поля поддерживают функции обоих уровней, например, интерфейсные функции поля *T* относятся к функциям уровня LLC, в то время как все остальные поля поддерживают функции уровня MAC. Существуют и еще три стандартных формата кадра Ethernet, описание которых можно найти на сайте книги.

(S) *Форматы кадров Ethernet*

Доступ к среде и передача данных

Метод доступа, используемый в сетях Ethernet на разделяемой проводной среде[1], носит название CSMA/CD (Carrier Sense Multiple Access with Collision Detection — прослушивание несущей частоты с множественным доступом и распознаванием коллизий). Название метода достаточно хорошо отражает его особенности. Все компьютеры в сети на разделяемой среде имеют возможность немедленно (с учетом задержки распространения сигнала в физической среде) получить данные, передаваемые любым из компьютеров в общую среду. Говорят, что среда, к которой подключены все станции, работает в режиме **коллективного доступа** (Multiply Access, MA). Чтобы получить возможность передавать кадр, интерфейс-отправитель должен убедиться, что разделяемая среда свободна. Это достигается прослушиванием основной гармоники сигнала, которая еще называется **несущей частотой** (Carrier Sense, CS). Признаком «незанятости» среды является отсутствие на ней несущей частоты, которая при манчестерском способе кодирования и тактовой частоте 10 МГц равна 5–10 МГц в зависимости от последовательности единиц и нулей, передаваемых в данный момент. Если среда свободна, то узел имеет право начать передачу кадра. В примере, показанном на рис. 10.5, узел 1 обнаружил, что среда свободна, и начал передавать свой кадр. В коаксиальном кабеле сигналы передатчика узла 1 распространяются в обе стороны, так что их получают все узлы сети. Кадр данных всегда сопровождается **преамбулой**, которая состоит из 7 байт, каждый из которых имеет значение 10101010, и 8-го байта, равного 10101011. Последний байт носит название **ограничителя начала кадра**. Преамбула нужна для вхождения приемника в побитовую и побайтовую синхронизацию с передатчиком. Наличие двух последовательных единиц

[1] В беспроводных сетях применяется другой метод доступа, известный как CSMA/CA. Этот метод рассматривается далее в разделе «Доступ к разделяемой среде» главы 25.

говорит приемнику о том, что преамбула закончилась и следующий бит является началом кадра.

Все станции, подключенные к кабелю, начинают записывать байты передаваемого кадра в свои внутренние буферы. Первые 6 байт кадра содержат адрес назначения. Та станция, которая узнает собственный адрес в заголовке кадра, продолжает записывать его содержимое в свой внутренний буфер, а остальные станции на этом прием кадра прекращают. Станция назначения обрабатывает полученные данные и передает их вверх по своему стеку. Кадр Ethernet содержит не только адрес назначения, но и адрес источника данных, поэтому станция-получатель знает, кому нужно послать ответ. Узел 2 во время передачи кадра узлом 1 также пытался начать передачу своего кадра, однако, обнаружив, что среда занята — на ней присутствует несущая частота, — вынужден ждать, пока узел 1 не прекратит передачу кадра.

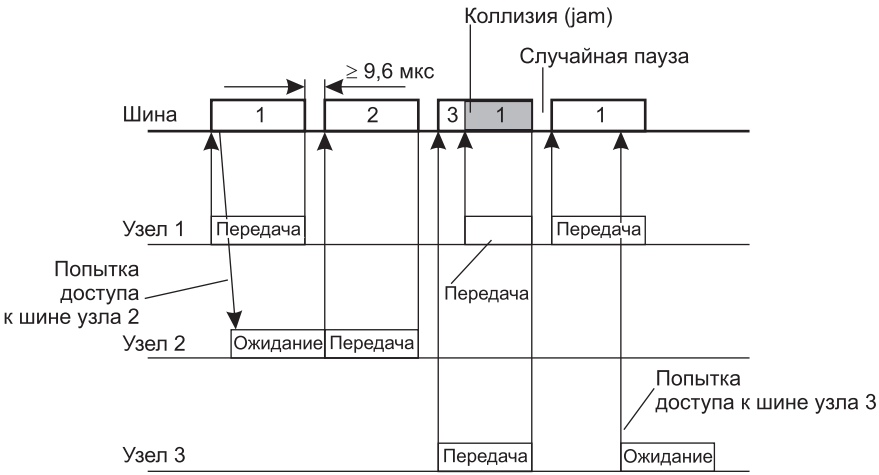


Рис. 10.5. Метод случайного доступа CSMA/CD

После окончания передачи кадра все узлы сети обязаны выдержать технологическую паузу, равную **межпакетному интервалу** (Inter Packet Gap, IPG) в 9,6 мкс. Эта пауза нужна для приведения сетевых адаптеров в исходное состояние, а также для предотвращения монопольного захвата среды одной станцией. После окончания технологической паузы узлы имеют право начать передачу своего кадра, так как среда свободна. В приведенном примере узел 2 дождался окончания передачи кадра узлом 1, сделал паузу в 9,6 мкс и начал передачу своего кадра.

Возникновение и распознавание коллизии

Механизм прослушивания среды и пауза между кадрами не гарантируют исключения ситуации, когда две или более станции одновременно решают, что среда свободна, и начинают передавать свои кадры. Говорят, что при этом происходит **коллизия**, так как содержимое обоих кадров сталкивается в общем кабеле и происходит искажение информации. Коллизия — это нормальная ситуация в работе сетей Ethernet на разделя-

емой среде. В примере на рис. 10.6 коллизию породила одновременная передача данных узлами 3 и 1. Для возникновения коллизии не обязательно, чтобы несколько станций начали передачу *абсолютно* одновременно, — напротив, более вероятна ситуация, когда один узел начинает передачу, а через некоторое (короткое) время другой узел, проверив среду и не обнаружив несущую (сигналы первого узла еще не успели до него дойти), начинает передачу своего кадра. Таким образом, возникновение коллизии — следствие распределения узлов сети в пространстве.

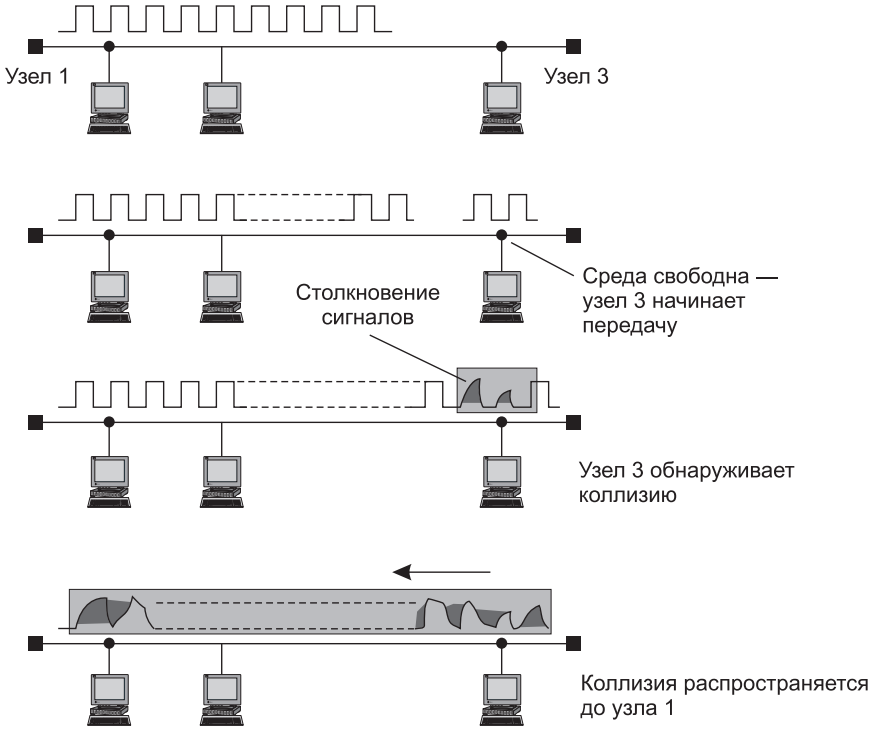


Рис. 10.6. Схема возникновения и распространения коллизии

Для корректной обработки коллизии все станции одновременно наблюдают за возникающими на кабеле сигналами. Если передаваемые и наблюдаемые сигналы отличаются, то фиксируется факт **обнаружения коллизии** (Collision Detection, CD). Для повышения вероятности скорейшего обнаружения коллизии всеми станциями сети станция, обнаружившая коллизию, прерывает передачу своего кадра (в произвольном месте, возможно, и не на границе байта), усугубляя коллизию посылкой в сеть последовательности из 32 бит, называемой **jam-последовательностью**. Затем обнаружившая коллизию передающая станция обязана прекратить передачу, сделать паузу в течение короткого случайного интервала времени, после чего снова предпринять попытку захвата среды и передачи кадра. Если 16 последовательных попыток передачи кадра вызывают коллизию, то передатчик должен прекратить попытки и отбросить этот кадр.

Поведение сети Ethernet при значительной нагрузке, когда коэффициент использования среды растет и начинает приближаться к 1, в целом соответствует графикам при анализе

модели теории очередей (см. главу 5). Рост времени ожидания освобождения среды в сетях Ethernet начинается раньше, чем в данной модели, вследствие того что в ней не учитывается такая особенность Ethernet, как коллизии.

Администраторы сетей Ethernet на разделяемой среде руководствовались простым эмпирическим правилом — коэффициент использования среды не должен превышать 30 %. Для поддержки чувствительного к задержкам трафика сети Ethernet (и другие сети на разделяемой среде) могут применять только один метод поддержания характеристик QoS — *недогруженный режим работы*. Надежное распознавание коллизий всеми станциями сети — необходимое условие корректной работы сети Ethernet. Для надежного распознавания коллизий должно выполняться соотношение:

$$T_{\min} \geq \text{RTT},$$

где $T_{\min}$ — время передачи кадра минимальной длины, а RTT — *время оборота*, то есть время, за которое сигнал, посланный некоторой станцией сети, доходит то точки коллизии и возвращается к станции отправителю в уже искаженной коллизией форме. В худшем случае сигнал должен пройти дважды между наиболее удаленными друг от друга станциями сети. При выполнении этого условия передающая станция должна успеть обнаружить коллизию, которую вызвал переданный ею кадр, еще до окончания передачи этого кадра. Очевидно, что выполнение условия зависит, с одной стороны, от минимальной длины кадра и скорости передачи данных протокола, а с другой — от длины кабельной системы сети и скорости распространения сигнала в кабеле (для разных типов кабеля эта скорость несколько отличается).

Все параметры протокола Ethernet, в том числе минимальный размер кадра, подобраны таким образом, чтобы при нормальной работе сети коллизии четко распознавались. Так, стандарт Ethernet определяет минимальную длину поля данных кадра в 46 байт (что вместе со служебными полями дает минимальную длину кадра 64 байта, а вместе с преамбулой — 72 байта, или 576 бит).

Отсюда может быть вычислено ограничение на расстояние между станциями. В стандарте Ethernet 10 Мбит/с время передачи кадра минимальной длины равно 575 битовым интервалам, следовательно, время оборота должно быть меньше 57,5 мкс. Расстояние, которое сигнал может пройти за это время, зависит от типа кабеля и для «толстого» коаксиального кабеля равно примерно 13 280 м. Учитывая, что за время 57,5 мкс сигнал должен пройти по линии связи дважды, расстояние между двумя узлами не должно быть больше 6635 м. В стандарте величина этого расстояния выбрана равной 2500 м, что существенно меньше. Это объясняется тем, что в сети могут использоваться повторители, которые нужны для соединения отдельных сегментов кабеля, внося задержки в распространение сигнала.

> Описанные соображения объясняют выбор минимальной длины поля данных кадра в 46 байт. Уменьшение этого значения до 0 привело бы к значительному сокращению максимальной длины сети. Несмотря на то что сегодня Ethernet не работает на разделяемой среде, требование к минимальной длине поля данных кадра Ethernet осталось в силе.

Требование $T_{\min} \geq \text{RTT}$ имеет одно интересное следствие: чем выше скорость протокола, тем меньше должна быть максимальная длина сети. Поэтому для Ethernet на разделяемой среде при скорости в 100 Мбит/с максимальная длина сети пропорционально уменьшается

до 250 м, а при скорости в 1 Гбит/с — до 25 м. Эта зависимость, наряду с резким ростом задержек при повышении загрузки сети, говорит о коренном недостатке как метода доступа CSMA/CD в частности, так и принципа разделяемой среды в целом.

Физические стандарты 10M Ethernet

При первоначальной стандартизации технологии Ethernet рабочей группой IEEE 802.3 был выбран вариант Ethernet на «толстом» коаксиальном кабеле, который получил название 10Base-5. Число 10 в этом названии обозначает номинальную битовую скорость передачи данных стандарта, то есть 10 Мбит/с, а слово «Base» — метод передачи на одной базовой частоте[1] (в данном случае — 10 МГц). Последний символ в названии стандарта физического уровня обозначает тип кабеля, в данном случае «5», отражает тот факт, что диаметр «толстого» коаксиала равен 0,5 дюйма.

> Данный подход к обозначению типа физического уровня Ethernet сохранился до настоящего времени, только вместо диаметра коаксиального кабеля в современных стандартах кодируется скорость, тип кабеля и некоторые другие параметры. Например, 1000Base-T определяет спецификацию скорости 1000 Мбит/с для витой пары, а 400GBase-LR – спецификацию скорости 400 Гбит/с на одномодовом оптоволоконном кабеле длиной до 10 км (LR — от Long Range).

В качестве метода кодирования сигналов был выбран манчестерский код.

Затем сети Ethernet на «толстом» коаксиальном кабеле были вытеснены сетями на более «тонком» коаксиале (диаметром 0,25 дюйма, что отражает название 10Base-2 этого стандарта), который позволял строить сети более экономичным способом.

Однако сети Ethernet на коаксиальном кабеле обладали одним существенным недостатком — отсутствием оперативной информации о состоянии кабеля и сложностью нахождения места его повреждения. Поэтому поиск неисправностей стал привычной процедурой и головной болью многочисленной армии сетевых администраторов коаксиальных сетей Ethernet.

Альтернатива появилась в середине 80-х, когда благодаря использованию витой пары и повторителей сети Ethernet стали гораздо более ремонтопригодными. К этому времени телефонные компании уже достаточно давно применяли многопарный кабель на основе неэкранированной витой пары для подключения телефонных аппаратов внутри зданий. Идея приспособить этот популярный вид кабеля для локальных сетей оказалась очень плодотворной — многие здания уже были оснащены нужной кабельной системой. Оставалось разработать способ подключения сетевых адаптеров и прочего коммуникационного оборудования к витой паре таким образом, чтобы минимизировать изменения в сетевых адаптерах и программном обеспечении сетевых операционных систем по сравнению с сетями Ethernet на коаксиале. Эта попытка оказалась успешной — переход на витую пару требует только замены приемника и передатчика сетевого адаптера, а метод доступа и все протоколы канального уровня остаются теми же, что и в сетях Ethernet на коаксиале. Результат реализации этой идеи — стандарт 10Base-T (T — от Twisted pair). Правда, для

[1] В отличие от методов, использующих несколько несущих частот, такие методы называются широкополосными и имеют в своем составе слово «broadband». Эти методы, хотя и были стандартизованы, не получили распространения в период популярности локальных сетей на разделяемой среде.

соединения узлов в сеть теперь обязательно требуется коммуникационное устройство — **многопортовый повторитель** Ethernet на витой паре.

Устройство повторителя схематично изображено на рис. 10.7. Каждый сетевой адаптер соединяется с повторителем двумя витыми парами: одна требуется для передачи данных от станции к повторителю (выход T_X сетевого адаптера), другая — для передачи данных от повторителя к станции (вход R_X сетевого адаптера). Повторитель побитно принимает сигналы от одного из конечных узлов и синхронно передает их на все свои остальные порты, исключая тот, с которого поступили сигналы, одновременно улучшая их электрические характеристики.

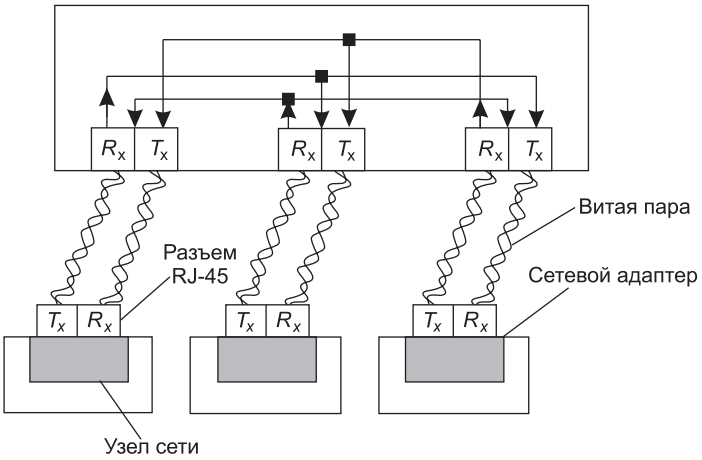


Рис. 10.7. Повторитель Ethernet на витой паре

Многопортовый повторитель часто называют **концентратором** или **хабом** (от англ. hub — центр, ступица колеса), так как в нем сконцентрированы соединения со всеми конечными узлами сети. Фактически хаб имитирует сеть на коаксиальном кабеле в том отношении, что физически отдельные отрезки кабеля на витой паре логически все равно представляют единую разделяемую среду. Все правила доступа к среде по алгоритму CSMA/CD сохраняются. При создании сети Ethernet на витой паре с большим числом конечных узлов хабы можно соединять друг с другом иерархически, образуя *древовидную структуру* (рис. 10.8). Добавление каждого хаба изменяет физическую структуру, оставляя без изменения логическую структуру сети. То есть независимо от числа хабов в сети сохраняется одна общая для всех интерфейсов разделяемая среда, так что передача кадра с любого интерфейса блокирует передатчики всех остальных интерфейсов.

Физическая структуризация сетей, построенных на основе витой пары, повысила надежность сетей Ethernet, поскольку в этом случае возможно контролировать состояние и локализовать отказы отдельных кабельных отрезков, подключающих конечные узлы к концентраторам. В случае обрыва, короткого замыкания или неисправности сетевого адаптера работа сети может быть быстро восстановлена путем отключения соответствующего сегмента кабеля. Но физическая структуризация не избавила Ethernet от ограничений по диаметру сети и количеству узлов, определяемых необходимостью распознавания коллизий. Дальнейший прогресс Ethernet требовал принципиально иного решения, которое нашлось в виде коммутируемого Ethernet.

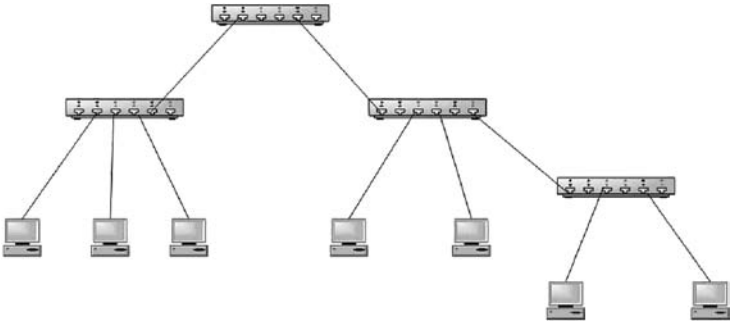


Рис. 10.8. Иерархическое соединение хабов

Коммутируемый Ethernet

Мост как предшественник и функциональный аналог коммутатора

Современные коммутаторы Ethernet являются наследниками мостов локальных сетей, которые широко использовались в сетях Ethernet и Token Ring на разделяемой среде. Более того, коммутаторы Ethernet по-прежнему функционально очень близки к вышедшим из употребления мостам, так как базовый алгоритм работы коммутатора и моста является одним и тем же и определяется одним стандартом IEEE 802.1D[1], хотя по традиции во всех новых стандартах IEEE, описывающих свойства коммутаторов, употребляется термин «мост», а не «коммутатор». Основное отличие коммутатора от моста состоит в большем количестве портов (мост, как правило, имел два порта, что и послужило поводом для его названия — мост между двумя сегментами) и более высокой производительности, достигаемой за счет одновременной передачи кадров между парами портов. Далее мы будем использовать эти термины как синонимы, выбирая нужный в зависимости от контекста.

Коммутаторы являются сегодня основным типом коммуникационных устройств, применяемых для построения локальных сетей. Их успех оказал решающее влияние на эволюцию Ethernet — в новых скоростных версиях Ethernet, начиная с версии 10G Ethernet, стандарт IEEE 802.3 описывает работу узлов только в коммутируемой среде.

Алгоритм прозрачного моста IEEE 802.1D

Мост локальной сети (LAN bridge), или просто **мост**, появился как средство построения крупных локальных сетей на разделяемой среде. Мост объединяет две или более разделяемые среды в единую сеть, при этом передача кадров между узлами каждой из объединяемых сред происходит по стандартным правилам изолированной разделяемой среды. Мост отвечает только за передачу кадров между объединенными средами, которые называются **сегментами локальной сети**.

---

[1] Комитет IEEE 802.1 занимается общими для всех технологий локальных сетей вопросами: работа моста, обеспечение качества обслуживания QoS и т. п.

Мост выполняет *логическую структуризацию сети*, то есть разделяет разделяемую среду на несколько сегментов и соединяет полученные сегменты, при этом мост не передает данные между сегментами побитно, как повторитель, а буферизует кадры и передает их затем в тот или иной сегмент (или сегменты) в зависимости от адреса назначения кадра (рис. 10.9).

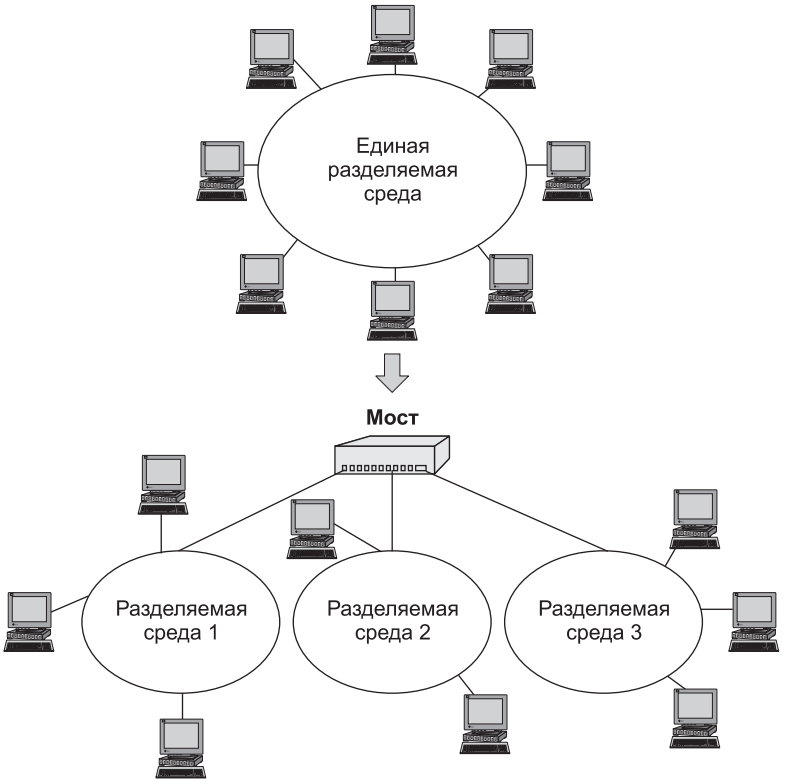


Рис. 10.9. Логическая структуризация сети

Нужно отличать логическую структуризацию от физической. Например, концентраторы стандарта 10Base-T позволяют построить сеть, состоящую из нескольких сегментов кабеля на витой паре, но это — физическая структуризация, так как логически все эти сегменты представляют собой единую разделяемую среду. Логическая структуризация сети с помощью мостов/коммутаторов является первым шагом на пути *виртуализации* сети, так как пользователям отдельного логического сегмента предоставляется виртуальный ресурс — коммуникационная среда с определенной пропускной способностью.

Логическая структуризация локальной сети позволяет решить несколько задач, основные из которых — повышение производительности и безопасности.

Повышение производительности сети, разделенной мостом на сегменты, происходит из-за того, что среда каждого сегмента разделяется теперь между меньшим числом конечных узлов. В примере на рис. 10.9 при разделении общей среды на три сегмента максимальное количество узлов, разделяющих среду, снизилось с 8 до 3. Нужно подчеркнуть, что мост принци-

пиально не исключает существования разделяемой среды: в каждом отдельном сегменте она по-прежнему может использоваться, и именно так строились сети в 80-х. Однако в современных коммутируемых локальных сетях применяется **микросегментация** — частный случай сегментации, когда сегмент состоит из одного узла, так что разделения среды не требуется.

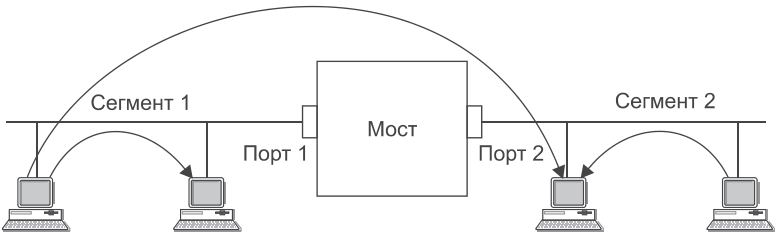
Устанавливая различные логические фильтры на мостах/коммутаторах, можно контролировать доступ пользователей к ресурсам других сегментов, чего не позволяют делать повторители. Так достигается *повышение безопасности данных*. Как мосты, так и коммутаторы продвигают кадры на основании одного и того же алгоритма, а именно — **алгоритма прозрачного моста**, описанного в стандарте IEEE 802.1D. Слово «прозрачный» в названии *алгоритм прозрачного моста* отражает тот факт, что конечные узлы сети функционируют, «не замечая» присутствия в сети мостов. Так как алгоритм прозрачного моста остался единственным актуальным алгоритмом мостов, то в дальнейшем мы будем опускать термин «прозрачный», подразумевая именно этот тип алгоритма работы моста/коммутатора.

Мост строит свою таблицу продвижения (адресную таблицу) на основании пассивного наблюдения за трафиком, циркулирующим в подключенных к его портам сегментах, учитывая адреса источников кадров данных, поступающих на его порты. По адресу источника кадра мост делает вывод о принадлежности узла-источника тому или иному сегменту сети.

ВНИМАНИЕ

Каждый порт моста работает как конечный узел своего сегмента, за одним исключением — порт моста может не иметь собственного MAC-адреса. Порты мостов не нуждаются в адресах для продвижения кадров, так как они работают в неразборчивом режиме захвата кадров, когда все поступающие на порт кадры, независимо от их адреса назначения, запоминаются на время в буферной памяти. Работая в неразборчивом режиме, мост «слушает» весь трафик, передаваемый в присоединенных к нему сегментах, и использует проходящие через него кадры для изучения топологии сети и построения таблицы продвижения. В том случае, когда порт моста/коммутатора имеет собственный MAC-адрес, он используется для целей, отличных от продвижения кадров, чаще всего — для удаленного управления портом; в этом случае порт представляет собой конечный узел сети и кадры протокола управления адресуются непосредственно ему.

Рассмотрим процесс автоматического создания таблицы продвижения моста и ее использования на примере простой сети, представленной на рис. 10.10.



MAC-адрес	Порт
1	1
2	1
3	2
4	2

Рис. 10.10. Принцип работы прозрачного моста/коммутатора

Мост соединяет два сетевых сегмента. Сегмент 1 составляют компьютеры, подключенные с помощью одного отрезка коаксиального кабеля к порту 1 моста, а сегмент 2 — компьютеры, подключенные с помощью другого отрезка коаксиального кабеля к порту 2 моста. В исходном состоянии мост не знает о том, компьютеры с какими MAC-адресами подключены к каждому из его портов. В этой ситуации мост просто передает любой захваченный и буферизованный кадр на *все* свои порты за исключением того порта, от которого этот кадр получен. В нашем примере у моста только два порта, поэтому он передает кадры с порта 1 на порт 2 и наоборот. Отличие работы моста в этом режиме от повторителя — в том, что он передает кадр, предварительно буферизуя его, а не бит за битом, как это делает повторитель. Буферизация отменяет логику работы всех сегментов как единой разделяемой среды. Когда мост собирается передать кадр с сегмента на сегмент, например, с сегмента 1 на сегмент 2, он, как обычный конечный узел, пытается получить доступ к разделяемой среде сегмента 2 по правилам алгоритма доступа, в данном примере — по правилам алгоритма CSMA/CD. Одновременно с передачей кадра на все порты мост изучает адрес источника кадра и делает запись о его принадлежности к тому или иному сегменту в своей **адресной таблице**. Эту таблицу также называют **таблицей фильтрации** или **продвижения**. Например, получив на порт 1 кадр от компьютера 1, мост делает первую запись в своей адресной таблице:

MAC-адрес 1 — порт 1.

Эта запись означает, что компьютер, имеющий MAC-адрес 1, принадлежит сегменту, подключенному к порту 1 коммутатора. Если все четыре компьютера данной сети проявляют активность и посылают друг другу кадры, то скоро мост построит полную адресную таблицу сети из четырех записей — по одной записи на узел (см. рис. 10.10). При каждом поступлении кадра на порт моста он прежде всего пытается найти адрес назначения кадра в адресной таблице. Продолжим рассмотрение действий моста на примере рис. 10.10:

1. При получении кадра, направленного от компьютера 1 компьютеру 3, мост просматривает адресную таблицу на предмет совпадения адреса в какой-либо из ее записей с адресом назначения — MAC-адресом 3. Запись с искомым адресом имеется в адресной таблице.
2. Далее мост выполняет второй этап анализа таблицы — проверяет, находятся ли компьютеры с адресами источника и назначения в одном сегменте. В примере компьютер 1 (MAC-адрес 1) и компьютер 3 (MAC-адрес 3) находятся в разных сегментах. Следовательно, мост выполняет операцию **продвижения** (forwarding) кадра — передает кадр на порт 2, ведущий в сегмент получателя, получает доступ к сегменту и передает туда кадр.
3. Если бы оказалось, что компьютеры принадлежат одному сегменту, то кадр просто был бы удален из буфера. Такая операция называется **фильтрацией** (filtering).
4. Если бы запись о MAC-адресе 3 отсутствовала в адресной таблице, то есть, другими словами, *адрес назначения был неизвестен* мосту, то он передал бы кадр на все свои порты, кроме порта — источника кадра, как и на начальной стадии процесса обучения.

Процесс обучения моста никогда не заканчивается и происходит одновременно с продвижением и фильтрацией кадров. Мост постоянно следит за адресами источника буферизуемых кадров, чтобы автоматически приспосабливаться к изменениям, происходящим

в сети, — перемещениям компьютеров из одного сегмента сети в другой, отключению и появлению новых компьютеров.

Входы адресной таблицы могут быть динамическими, создаваемыми в процессе *самообучения* моста, и статическими, создаваемыми *вручную* администратором сети. **Статические записи** не имеют срока жизни, что дает администратору возможность влиять на работу моста, например, ограничивая передачу кадров с определенными адресами из одного сегмента в другой.

Динамические записи имеют срок жизни — при создании или обновлении записи в адресной таблице с ней связывается отметка времени. По истечении определенного тайм-аута запись помечается как недействительная, если за это время мост не принял ни одного кадра с данным адресом в поле адреса источника. Это дает возможность мосту автоматически реагировать на перемещения компьютера из сегмента в сегмент — при его отключении от старого сегмента запись о принадлежности компьютера к этому сегменту со временем вычеркивается из адресной таблицы. После подключения компьютера к другому сегменту его кадры начнут попадать в буфер моста через другой порт, а в адресной таблице появится новая запись, соответствующая текущему состоянию сети.

Кадры с широковещательными и групповыми MAC-адресами, как и кадры с неизвестными адресами назначения, передаются мостом на все его порты. Такой режим распространения кадров называется **затоплением сети** (flooding). Наличие мостов в сети не препятствует распространению широковещательных и групповых кадров по всем сегментам сети. Однако это является достоинством только тогда, когда такой адрес выработан корректно работающим узлом. Нередко в результате каких-либо программных или аппаратных сбоев протокол верхнего уровня либо сетевой адаптер начинают работать некорректно, а именно постоянно с высокой интенсивностью генерировать кадры с широковещательным адресом. Мост в соответствии со своим алгоритмом передает ошибочный трафик во все сегменты. Такая ситуация называется **широковещательным штормом** (broadcast storm).

К сожалению, мосты не защищают сети от широковещательного шторма, во всяком случае, по умолчанию, как это делают маршрутизаторы (вы познакомитесь с этим свойством маршрутизаторов в части IV). Максимум, что может сделать администратор с помощью коммутатора для борьбы с широковещательным штормом — установить для каждого порта моста предельно допустимую интенсивность передачи кадров с широковещательным адресом. Но при этом нужно точно знать, какая интенсивность является нормальной, а какая — ошибочной. При смене протоколов ситуация в сети может измениться, и то, что вчера считалось ошибочным, сегодня может оказаться нормой. На рис. 10.11 показана типичная структура моста. Функции доступа к среде при приеме и передаче кадров выполняют микросхемы MAC, которые идентичны микросхемам сетевого адаптера.

Протокол, реализующий алгоритм коммутатора, располагается между уровнями MAC и LLC. На рис. 10.12 показана копия экрана терминала с адресной таблицей моста.

Из выводимой на экран адресной таблицы видно, что сеть состоит из двух сегментов — LAN *A* и LAN *B*. В сегменте LAN *A* имеются по крайней мере 3 станции, в сегменте LAN *B* — 2 станции. Четыре адреса, помеченные звездочками, являются статическими, то есть назначенными администратором вручную. Адрес, помеченный плюсом, является динамическим адресом с истекшим сроком жизни.

Таблица имеет поле *Dispn* — «disposition» (это «распоряжение» мосту о том, какую операцию нужно проделать с кадром, имеющим данный адрес назначения). Обычно при автоматическом составлении таблицы в этом поле ставится условное обозначение порта назначения, при ручном же задании адреса в это поле можно внести нестандартную операцию обработки кадра. Например, операция *Flood* (затопление) заставляет мост распространять кадр в широковещательном режиме, несмотря на то что его адрес назначения не является широковещательным. Операция *Discard* (отбросить) говорит мосту, что кадр с таким адресом не нужно передавать на порт назначения. Вообще говоря, операции, задаваемые в поле *Dispn*, определяют особые условия фильтрации кадров, дополняющие стандартные условия их распространения. Такие условия обычно называют **пользовательскими фильтрами** (подробнее см. раздел «Фильтрация трафика» главы 11).

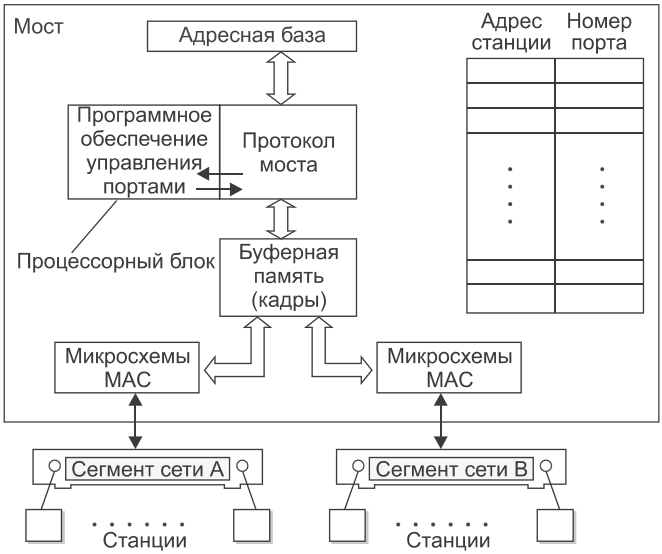


Рис. 10.11. Структура моста/коммутатора

Forwarding Table Page 1 of 1

Address	Dispn	Address	Dispn	Address	Dispn
00608CB17E58	LAN B	0000810298D6	LAN A	02070188ACA	LAN A
00008101C4DF	LAN B	+ 000081016A52	LAN A	* 010081000100	Flood
* 010081000101	Discard	* 0180C2000000	Discard	* 000081FFD166	Flood

Статус адреса: срок жизни записи истек

Exit Next Page Prev Page Edit Table Search Item Go Page

+ Unlearned * Static Total Entries = 9 Static Entries = 4

Use cursor keys to choose option. Press <RETURN> to select.

Press <CTRL> <P> to return to Main Menu

Рис. 10.12. Адресная таблица моста/коммутатора

Топологические ограничения при применении мостов в локальных сетях

Серьезным ограничением функциональных возможностей мостов и коммутаторов является отсутствие поддержки петлеобразных конфигураций сети.

Рассмотрим это ограничение на примере сети, показанной на рис. 10.13.

Два сегмента Ethernet параллельно соединены двумя мостами, так что образовалась петля. Пусть новая станция с MAC-адресом 123 впервые начинает работу в данной сети. Обычно начало работы любой ОС сопровождается рассылкой широковещательных кадров, в которых станция заявляет о своем существовании и одновременно ищет сервер сети.

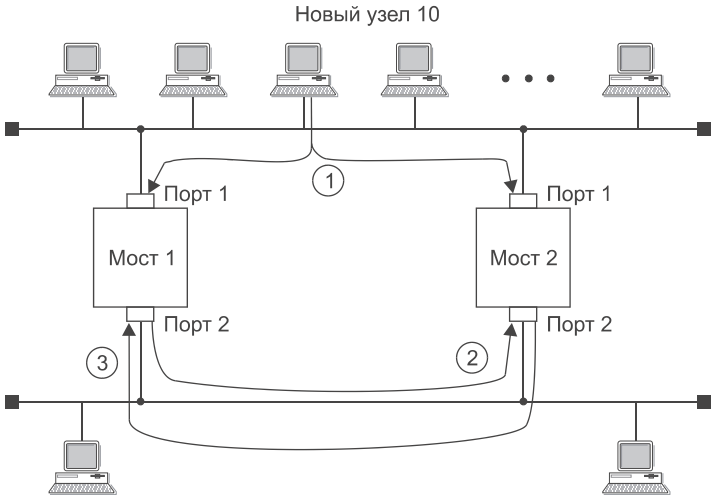


Рис. 10.13. Влияние замкнутых маршрутов на работу коммутаторов

На этапе 1 станция посылает первый кадр с широковещательным адресом назначения и адресом источника 123 в свой сегмент. Кадр попадает как в мост 1, так и в мост 2. В обоих мостах новый адрес источника 123 заносится в адресную таблицу с пометкой о его принадлежности сегменту 1, то есть создается новая запись вида:

MAC-адрес 123 — Порт 1

Так как адрес назначения широковещательный, каждый мост должен передать кадр на сегмент 2. Эта передача происходит поочередно в соответствии с методом случайного доступа технологии Ethernet. Пусть первым доступ к сегменту 2 получает мост 1 (этап 2 на рис. 10.13). При появлении кадра на сегменте 2 мост 2 принимает его в свой буфер и обрабатывает. Он видит, что адрес 123 уже есть в его адресной таблице, но пришедший кадр является более свежим, и он решает, что адрес 123 принадлежит сегменту 2, а не 1. Поэтому мост 2 корректирует содержимое базы и делает запись о том, что адрес 123 принадлежит сегменту 2:

MAC-адрес 123 — Порт 2

Аналогично поступает мост 1, когда мост 2 передает свою копию кадра на сегмент 2. Перечислим последствия наличия петли в сети:

- «Размножение» кадра, то есть появление нескольких его копий (в данном случае — двух, но если бы сегменты были соединены тремя мостами — трех и т. д.).
- Бесконечная циркуляция обеих копий кадра по петле в противоположных направлениях, а значит, засорение сети ненужным трафиком.
- Постоянная перестройка мостами своих адресных таблиц, так как кадр с адресом источника 123 будет появляться то на одном порту, то на другом.

В целях исключения всех этих нежелательных эффектов мосты/коммутаторы нужно применять так, чтобы между логическими сегментами не было петель, то есть строить с помощью коммутаторов только древовидные структуры, гарантирующие наличие единственного пути между любыми двумя сегментами. Тогда кадры от каждой станции будут поступать на мост/коммутатор всегда с одного и того же порта, и коммутатор сможет правильно решать задачу выбора рационального маршрута в сети.

В небольших сетях сравнительно легко гарантировать наличие одного и только одного пути между двумя сегментами. Но когда количество соединений возрастает, вероятность непреднамеренного образования петли оказывается высокой.

Возможна и другая причина возникновения петель. Так, для повышения надежности желательно иметь между мостами/коммутаторами резервные связи, которые не участвуют в нормальной работе основных связей по передаче информационных кадров станций, но при отказе какой-либо основной связи образуют новую связную рабочую конфигурацию без петель. Избыточные связи необходимо блокировать, то есть переводить их в неактивное состояние. В сетях с простой топологией эта задача решается вручную путем блокирования соответствующих портов мостов/коммутаторов. В больших сетях со сложными связями используются алгоритмы, которые позволяют решать задачу обнаружения петель автоматически. Наиболее известным из них является стандартный **алгоритм покрывающего дерева** (Spanning Tree Algorithm, STA), который детально рассмотрен в главе 11.

Коммутаторы

Параллельная коммутация

При появлении на рубеже 1980-х — 1990-х годов быстрых протоколов, производительных персональных компьютеров, мультимедийной информации и разделении сети на большое количество сегментов классические *мосты* перестали справляться с работой. Обслуживание потоков кадров уже несколькими портами с помощью одного процессорного блока требовало значительного повышения быстродействия процессора, что довольно дорого.

Более эффективным оказалось решение, которое и «породило» коммутаторы: для обслуживания потока, поступающего на каждый порт, в устройство ставился отдельный специализированный процессор, который реализовывал алгоритм прозрачного моста. По сути, коммутатор — это мультипроцессорный мост, способный параллельно продвигать кадры сразу между всеми парами своих портов. Но если при добавлении процессорных блоков компьютер не перестали называть компьютером, а добавили только прилагательное «мультипроцессорный», то с мультипроцессорными мостами произошла метаморфоза — во многом по маркетинговым причинам они превратились в коммутаторы. Нужно отметить,

что помимо процессоров портов коммутатор имеет центральный процессор, который координирует работу портов, отвечая за построение общей таблицы продвижения, а также поддерживая функции конфигурирования и управления коммутатором.

Со временем коммутаторы вытеснили из локальных сетей классические однопроцессорные мосты. Основная причина этого — существенно более высокая производительность, с которой коммутаторы передают кадры между сегментами сети. Если мосты могли даже замедлять работу сети, то коммутаторы всегда выпускаются с процессорами портов, способными передавать кадры с той максимальной скоростью, на которую рассчитан протокол. Добавление к этому возможности параллельной передачи кадров между портами предопределило судьбу и мостов, и коммутаторов.

Производительность коммутаторов на несколько порядков выше, чем мостов, — коммутаторы могут передавать до нескольких десятков, а иногда и сотен миллионов кадров в секунду, в то время как мосты обычно обрабатывали 3–5 тысяч кадров в секунду.

За время своего существования уже без конкурентов-мостов коммутаторы вобрали в себя многие дополнительные функции, родившиеся в результате естественного развития сетевых технологий. К этим функциям относятся, например, поддержка виртуальных сетей (VLAN), агрегирование линий связи, приоритизация трафика и т. п. Развитие технологии производства заказных микросхем также способствовало успеху коммутаторов. В результате процессоры портов сегодня обладают такой вычислительной мощностью, которая позволяет им быстро реализовывать весьма сложные алгоритмы обработки трафика, например, выполнять его классификацию и профилирование.

Основной причиной повышения производительности сети при использовании коммутатора является *параллельная* обработка нескольких кадров.

Этот эффект иллюстрирует рис. 10.14, на котором показана идеальная в отношении производительности ситуация, когда четыре порта из восьми передают данные с максимальной для протокола Ethernet скоростью в 10 Мбит/с. Причем они передают эти данные на остальные четыре порта коммутатора, не конфликтуя: потоки данных между узлами сети распределились так, что для каждого принимающего кадры порта есть свой выходной порт. Если коммутатор успевает обрабатывать входной трафик при максимальной интенсивности поступления кадров на входные порты, то общая производительность коммутатора в приведенном примере составит 4 × 10 = 40 Мбит/с, а при обобщении примера для N портов — $(N/2) \times 10$ Мбит/с. В таком случае говорят, что *коммутатор предоставляет каждой станции или сегменту, подключенному к его портам, выделенную пропускную способность протокола.*

Естественно, что в сети не всегда складывается описанная ситуация. Если двум станциям, например, станциям, подключенным к портам 3 и 4, одновременно нужно записывать данные на один и тот же сервер, подключенный к порту 8, то коммутатор не сможет выделить каждой станции по 10 Мбит/с, так как порт 8 не в состоянии передавать данные со скоростью 20 Мбит/с. Кадры станций будут ожидать во внутренних очередях входных портов 3 и 4, когда освободится порт 8 для передачи очередного кадра. Очевидно, хорошим решением для такого распределения потоков данных было бы подключение сервера к более высокоскоростному порту, например Fast Ethernet или Gigabit Ethernet.

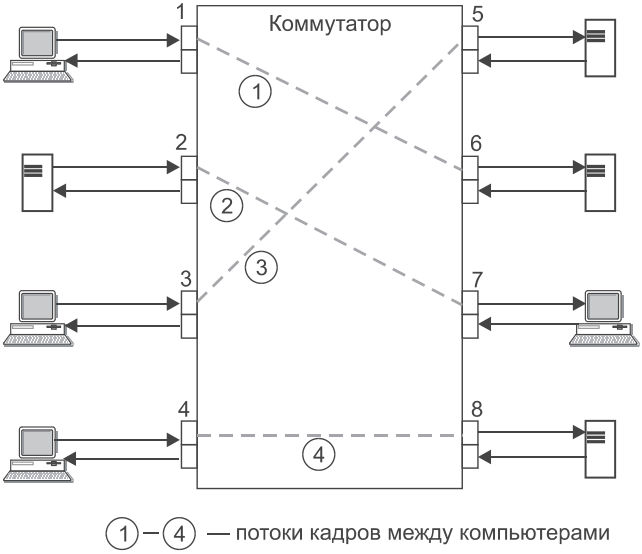


Рис. 10.14. Параллельная передача кадров коммутатором

Дуплексный режим работы

Технология коммутации сама по себе не имеет непосредственного отношения к методу доступа к среде, который используется портами коммутатора. При подключении к порту коммутатора сегмента, представляющего собой разделяемую среду, данный порт, как и все остальные узлы такого сегмента, должен поддерживать полудуплексный режим. Однако когда к каждому порту коммутатора подключен не сегмент, а только *один* компьютер, причем по двум физически раздельным каналам, как это происходит почти во всех стандартах Ethernet, кроме коаксиальных версий Ethernet, ситуация становится не такой однозначной. Порт может работать как в обычном полудуплексном режиме, так и в дуплексном.

В **полудуплексном режиме** работы порт коммутатора по-прежнему распознает коллизии. Доменом коллизий в этом случае является участок сети, включающий передатчик и приемник коммутатора, передатчик и приемник сетевого адаптера компьютера, а также две витые пары, соединяющие передатчики с приемниками. Коллизия возникает, когда передатчики порта коммутатора и сетевого адаптера одновременно или почти одновременно начинают передачу своих кадров.

В **дуплексном режиме** одновременная передача данных передатчиком порта коммутатора и сетевого адаптера коллизией не считается — этот режим работы может рассматриваться в качестве штатного для отдельных дуплексных каналов передачи данных, ранее использовавшегося в протоколах глобальных сетей. При дуплексной связи порты Ethernet стандарта 10 Мбит/с могут передавать данные со скоростью 20 Мбит/с — по 10 Мбит/с в каждом направлении.

Долгое время коммутаторы Ethernet сосуществовали в локальных сетях с концентраторами Ethernet: на концентраторах строились нижние уровни сети здания (в частности, сети рабочих групп и отделов), а коммутаторы служили для объединения этих сегментов в общую сеть. Но постепенно коммутаторы стали применяться и на нижних этажах, вы-

тесняя концентраторы, так как цены на коммутаторы постоянно снижались, а их производительность росла (за счет поддержки более скоростных версий технологии Ethernet, то есть Fast Ethernet со скоростью 100 Мбит/с, Gigabit Ethernet со скоростью 1 Гбит/с, 10G Ethernet со скоростью 10 Гбит/с и 100G Ethernet со скоростью 100 Гбит/с — эти версии рассматриваются далее в разделе «Скоростные версии Ethernet»). Этот процесс завершился вытеснением концентраторов Ethernet и переходом к полностью коммутируемым сетям (рис. 10.15).

В полностью коммутируемой сети Ethernet все порты работают в дуплексном режиме, а продвижение кадров осуществляется на основе MAC-адресов.

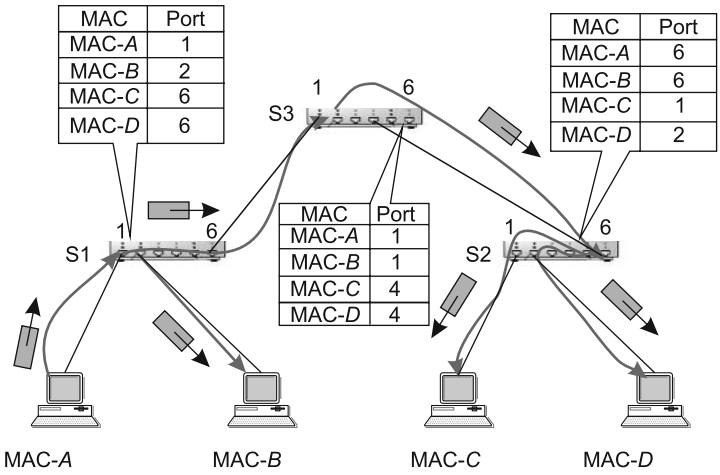


Рис. 10.15. Полностью коммутируемая сеть Ethernet

При разработке технологий Fast Ethernet и Gigabit Ethernet дуплексный режим стал одним из двух полноправных стандартных режимов работы узлов сети. Однако практика применения первых коммутаторов с портами Gigabit Ethernet показала, что они практически всегда применяются в дуплексном режиме для взаимодействия с другими коммутаторами или высокоскоростными сетевыми адаптерами. Поэтому при разработке версий стандартов 10G и 100G Ethernet его разработчики не стали создавать версию для работы в полудуплексном режиме, окончательно закрепив уход разделяемой среды из технологии Ethernet.

Неблокирующие коммутаторы

Высокая производительность — одно из главных достоинств коммутаторов. С понятием производительности тесно связано понятие неблокирующего коммутатора.

> Коммутатор называют **неблокирующим**, если он может передавать кадры через свои порты с той же скоростью, с которой они на них поступают.

Под коммутатором, способным поддерживать *устойчивый неблокирующий режим работы*, подразумевается коммутатор, передающий кадры со скоростью их поступления в течение

произвольного промежутка времени. Для поддержания подобного режима нужно распределить потоки кадров по выходным портам таким образом, чтобы, во-первых, порты справлялись с нагрузкой и, во-вторых, коммутатор мог всегда в среднем передать на выходы столько кадров, сколько их поступило на входы. Если же входной поток кадров (просуммированный по всем портам) в среднем будет превышать выходной поток кадров (также просуммированный по всем портам), то кадры будут накапливаться в буферной памяти коммутатора и при переполнении — просто отбрасываться.

Для поддержания устойчивого неблокирующего режима работы коммутатора необходимо, чтобы его производительность удовлетворяла условию $C_k = (\Sigma C_{pi})/2$, где C_k — производительность коммутатора, C_{pi} — максимальная производительность протокола, поддерживаемого i-м портом коммутатора.

В этом соотношении под производительностью коммутатора в целом понимается его способность продвигать на передатчики всех своих портов определенное количество кадров, принимаемых от приемников всех своих портов. В суммарной производительности портов каждый проходящий кадр учитывается дважды (как входящий и как выходящий), а так как в устойчивом режиме входной трафик равен выходному, то минимально достаточная производительность коммутатора для поддержки неблокирующего режима равна половине суммарной производительности портов. Так, если порт стандарта Ethernet со скоростью 10 Мбит/с работает в полудуплексном режиме, то производительность порта C_{pi} равна 10 Мбит/с, а если в дуплексном — то 20 Мбит/с. Иногда говорят, что коммутатор поддерживает *мгновенный неблокирующий режим*. Это означает, что он может принимать и обрабатывать кадры от всех своих портов на максимальной скорости протокола независимо от того, обеспечиваются ли условия устойчивого равновесия между входным и выходным трафиком. Но обработка некоторых кадров при этом может быть неполной — при занятости выходного порта кадр помещается в буфер коммутатора.

Для поддержки мгновенного неблокирующего режима коммутатор должен обладать большей собственной производительностью, а именно — она должна быть равна суммарной производительности его портов: $C_k = \Sigma C_{pi}$.

Приведенные соотношения справедливы для портов с любыми скоростями, то есть портов стандартов Ethernet со скоростью 10 Мбит/с, Fast Ethernet, Gigabit Ethernet, 10G и 100G Ethernet.

Способы, которыми поддерживается способность коммутатора поддерживать неблокирующий режим, могут быть разными. Необходимым требованием является способность процессора порта обрабатывать потоки кадров с максимальной для физического уровня этого порта скоростью. Так, максимальная производительность порта Ethernet стандарта 10 Мбит/с равна 14 880 кадров (минимальной длины) в секунду. Это означает, что процессоры портов Ethernet стандарта 10 Мбит/с неблокирующего коммутатора должны поддерживать продвижение кадров со скоростью 14 880 кадров в секунду. Как мы увидим дальше, более скоростные версии Ethernet сохраняют формат кадра Ethernet, сокращая межкадровый интервал пропорционально увеличению битовой скорости версии (то есть в 10 раз при повышении скорости в 10 раз). Поэтому максимальные значения скорости продвижения кадров также растут пропорционально росту битовой скорости: например, Fast Ethernet обеспечивает максимальную скорость продвижения в 148 800 кадров в секунду,

а Gigabit Ethernet — в 1 488 000 кадров в секунду. Соответственно, должна расти и скорость продвижения коммутатора Ethernet с высокоскоростными портами.

Однако только адекватной производительности процессоров портов недостаточно для того, чтобы коммутатор был неблокирующим. Необходимо, чтобы достаточной производительностью обладали все элементы архитектуры коммутатора, включая центральный процессор, общую память, шины, соединяющие отдельные модули между собой, саму архитектуру коммутатора (наиболее распространенные архитектуры коммутаторов мы рассмотрим позже). В принципе, задача создания неблокирующего коммутатора аналогична задаче создания высокопроизводительного компьютера — и та и другая решаются комплексно: за счет соответствующей архитектуры объединения модулей в едином устройстве и адекватной производительности каждого отдельного модуля устройства. Для ускорения операций коммутации сегодня во всех коммутаторах используются заказные специализированные БИС — ASIC (Application-Specific Integrated Circuit), которые оптимизированы для выполнения основных операций коммутации. Часто в одном коммутаторе имеется несколько специализированных БИС, каждая из которых выполняет функционально законченную часть операций.

Важную роль в построении коммутаторов играют и программируемые микросхемы **FPGA** (Field-Programmable Gate Array — программируемый в условиях эксплуатации массив вентилей). Эти микросхемы могут выполнять все функции, которые выполняют микросхемы ASIC, но, в отличие от последних, могут программироваться и перепрограммироваться производителями коммутаторов (и даже пользователями). Это свойство позволило резко удешевить процессоры портов коммутаторов, выполняющих сложные операции, например, профилирование трафика, так как производитель FPGA выпускает свои микросхемы массово, а не по заказу того или иного производителя оборудования. Кроме того, применение микросхем FPGA позволяет производителям коммутаторов оперативно вносить изменения в логику работы порта при появлении новых стандартов или изменении действующих.

В коммутаторах также применяются **сетевые процессоры** (Network Processor Unit, **NPU**). Этот тип процессоров имеет набор команд, ориентированных на обработку пакетов, обеспечивая еще бóльшую гибкость, чем микросхемы FPGA.

Помимо процессорных микросхем для успешной неблокирующей работы коммутатору нужно иметь быстродействующий *узел обмена*, предназначенный для передачи кадров между процессорными микросхемами портов. В настоящее время в коммутаторах узел обмена строится на основе одной из трех схем:

- *Коммутационная матрица.* Такая матрица состоит из двоичных переключателей, которые выполняют коммутацию канала между парой портов на время передачи данных пакета. Это наиболее простое решение, но работает оно только в случае фиксированного количества портов коммутатора: добавление портов требует изменения организации матрицы.
- *Общая шина.* Это наиболее традиционный и гибкий метод объединения модулей вычислительного устройства, широко применяемый в компьютерах (шина PCI настольных компьютеров является наиболее известным примером).
- *Разделяемая многовходовая память.* В памяти для каждого порта организуется отдельная очередь пакетов. Любой порт может поместить пришедший пакет в эту очередь, а порт, для которого очередь предназначена, выбирает из нее пакеты и передает в сеть. Для поддержания нужной скорости работы коммутатора разделяемая память должна обладать высоким быстродействием.

Иногда эти схемы комбинируются в одном коммутаторе, например, коммутационная матрица, обслуживающая определенное количество портов, и общая шина, обеспечивающая взаимодействие между группами портов, обслуживаемых разными коммутационными матрицами.

ПО коммутаторов, реализующее алгоритм прозрачного моста и обеспечивающее его конфигурирование и управление, долгое время оставалось фирменным, так что администраторам сетей приходилось приобретать новые навыки при переходе на оборудование другого производителя. Однако в последнее время усиливается другая тенденция — все большую популярность завоевывают **открытые коммутаторы**, называемые в англоязычном мире **Bare Metal** switches или **White Box** switches. В таких коммутаторах аппаратная часть отделена от программной, причем в качестве последней используется одна из сетевых ОС с открытым исходным кодом, чаще всего — одна из специализированных для сетевых операций версий Unix/Linux. В случае «голого железа», то есть коммутатора типа Bare Metal, вы приобретаете коммутатор без предустановленной ОС, но с уже установленным ее загрузчиком. Загрузчик ОС у такого коммутатора работает по открытому стандарту ONIE (Open Network Install Environment), которого придерживаются разработчики открытых сетевых систем, что позволяет установить на коммутатор любую из имеющихся ОС. Коммутаторы типа White Box имеют ту же архитектуру, что и коммутаторы Bare Metal, но продаются с уже установленной открытой ОС, так что потребитель может сразу же включить коммутатор и начать его конфигурировать без необходимости проходить через этап установки сетевой ОС (в последующем он может заменить установленную ОС на другую).

Скоростные версии Ethernet

Скорость 10 Мбит/с первой стандартной версии Ethernet долгое время удовлетворяла потребности пользователей локальных сетей. Однако в начале 90-х стала ощущаться недостаточная пропускная способность Ethernet, так как скорость обмена с сетью стала существенно меньше скорости внутренней шины компьютера. Кроме того, начали появляться новые мультимедийные приложения, гораздо более требовательные к скорости сети, чем их текстовые предшественники. В поисках решения проблемы ведущие производители сетевого оборудования начали интенсивные работы по повышению скорости Ethernet при сохранении главного достоинства этой технологии — простоты и невысокой стоимости оборудования. Результатом стало появление новых скоростных стандартов Ethernet: Fast Ethernet (скорость 100 Мбит/с), Gigabit Ethernet (1000 Мбит/с, или 1 Гбит/с), 10G Ethernet (10 Гбит/с), 100G Ethernet, 200G Ethernet (200 Гбит/с), 400G и 800G Ethernet.

Разработчикам новых скоростных стандартов Ethernet удалось сохранить основные черты классической технологии Ethernet и прежде всего простой способ обмена кадрами без встраивания в технологию сложных контрольных процедур. Этот фактор оказался решающим в соревновании технологий локальных сетей, так как выбор пользователей всегда склонялся в сторону простого наращивания скорости сети, а не в сторону решений, связанных с более эффективным расходованием той же самой пропускной способности с помощью более сложной и дорогой технологии.

Значительный вклад в «победу» Ethernet внесли также коммутаторы локальных сетей, так как их успех привел к отказу от разделяемой среды, где технология Ethernet всегда была

уязвимой из-за случайного характера метода доступа. Начиная с версии 10G Ethernet разработчики перестали обеспечивать обратную совместимость с предыдущими вариантами и отказались от поддержки разделяемой среды. Повышение скорости работы Ethernet было достигнуто за счет нескольких факторов:

- улучшение качества кабелей, применяемых в компьютерных сетях;
- совершенствование методов кодирования данных;
- использование параллельных потоков данных.

Все отличия скоростных версий Ethernet от классической версии 10M Ethernet проявляются на физическом уровне (рис. 10.16). Уровни MAC и LLC в Fast Ethernet остались абсолютно теми же, и их описывают прежние главы стандартов 802.3 и 802.2. Поэтому, рассматривая очередную версию технологии Ethernet, будем изучать только вариант организации ее физического уровня.

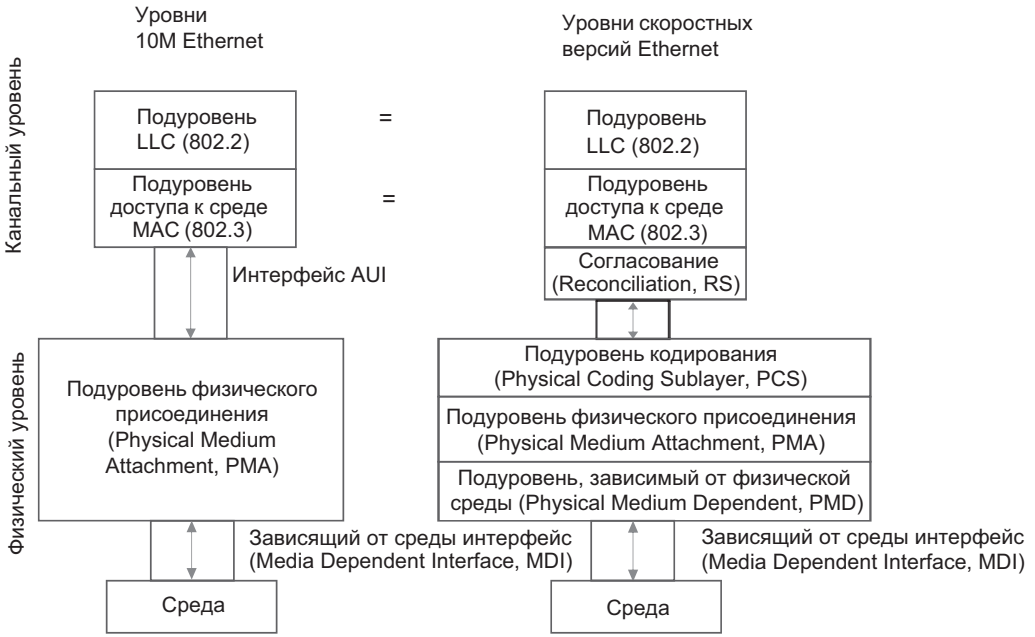


Рис. 10.16. Структура физического уровня версии 10M Ethernet и скоростных версий Ethernet

Организация физического уровня скоростных версий Ethernet является модульной. Модульность обеспечивает гибкость физического уровня Ethernet, когда путем замены некоторого модуля можно обеспечить поддержку различных методов кодирования данных или различных сред распространения сигнала. Физический уровень Fast Ethernet состоит из следующих модулей:

- **Независимый от среды интерфейс** (x Media Independent Interface, xMII). Этот интерфейс поддерживает независимый от физической среды способ обмена данными между

подуровнями MAC и PHY. Обозначение «x» отражает тот факт, что для различных скоростей передачи данных этот интерфейс имеет свою специфику, например, для скорости 1 Гбит/с применяется интерфейс GMII, а для скорости 10 Гбит/с — интерфейс XGMII.

- **Модуль согласования** (Reconciliation) нужен для того, чтобы уровень MAC, рассчитанный ранее на интерфейс AUI, мог работать с физическим уровнем через интерфейс xMII.
- **Устройство физического уровня** (Physical Layer Device, PHY) состоит, в свою очередь, из трех подуровней:
  - подуровень кодирования данных (Physical Coding Sublayer, **PCS**), преобразующий поступающие от уровня MAC байты в символы логического кода, например 4B/5B или 64B/66B;
  - подуровень физического присоединения (Physical Media Attachment, **PMA**), преобразующий параллельный код в поток битов и восстанавливающий синхронизацию потока;
  - подуровень зависимости от физической среды (Physical Media Dependent, **PMD**), который обеспечивает формирование сигналов нужной мощности на физической линии связи в соответствии с выбранным методом физического кодирования, например NRZI, PAM5 или PAM**4**.

Одной из функций подуровня кодирования данных PCS является функция автопереговоров.

Функция автопереговоров позволяет двум физически соединенным устройствам, поддерживающим несколько стандартов физического уровня, отличающихся битовой скоростью, согласовать наиболее выгодный режим работы. Обычно процедура автопереговоров происходит при подсоединении сетевого адаптера компьютера к порту коммутатора или маршрутизатора в том случае, когда оба устройства могут работать на нескольких скоростях — например, на скоростях 100 Мбит/с, 1000 Мбит/с и 10 Гбит/с. Режим с более высокой скоростью имеет более высокий приоритет. Если некоторый режим может работать как в полудуплексном, так и в полнодуплексном вариантах (это относится к скоростям менее 10 Гбит/с), то полнодуплексный режим имеет более высокий приоритет по сравнению с полудуплексным.

Переговорный процесс происходит при включении питания устройства, а также может быть инициирован в любой момент модулем управления устройства. Устройство, начавшее процесс автопереговоров, посылает своему партнеру пачку специальных импульсов **FLP** (Fast Link Pulse), содержащих 8-битное слово, кодирующее предлагаемый режим взаимодействия, начиная с самого приоритетного, поддерживаемого данным узлом. Импульсы FLP имеют длительность 100 нс, как и импульсы LIT, используемые для тестирования целостности физического соединения в стандарте 10Base-T, но вместо передачи одного импульса LIT через каждые 16 мс здесь через тот же интервал передается пачка импульсов FLP. Если узел-партнер имеет функцию автопереговоров и также способен поддерживать предложенный режим, то отвечает пачкой импульсов FLP, подтверждающей этот режим, и на этом переговоры заканчиваются. Если узел-партнер не может поддерживать запрошенный режим, то указывает в своем ответе имеющийся в его распоряжении следующий по степени приоритетности режим, и этот режим выбирается в качестве рабочего.

Fast Ethernet

Спецификация Fast Ethernet была первой из серии спецификации скоростных версий Ethernet, поэтому она удостоилась названия «быстрой» (Fast). Разработчикам технологии Fast Ethernet удалось обеспечить ее преемственность с классической технологией Ethernet 10 Мбит/с. Fast Ethernet поддерживает *три* варианта физической среды:

- волоконно-оптический многомодовый кабель (два волокна);
- витая пара категории 5 (две пары);
- витая пара категории 3 (четыре пары).

Заметим, что коаксиальный кабель, давший миру первую сеть Ethernet, в число разрешенных сред передачи данных технологии Fast Ethernet не попал.

Официальный стандарт 802.3 установил три различные спецификации для физического уровня Fast Ethernet и дал им следующие названия:

- **100Base-TX** — для двухпарного кабеля на неэкранированной витой паре UTP категории 5;
- **100Base-T4** — для четырехпарного кабеля на неэкранированной витой паре UTP категории 3, 4 или 5;
- **100Base-FX** — для многомодового оптоволоконного кабеля с двумя волокнами.

Для всех трех стандартов справедливы перечисленные далее утверждения и характеристики.

> Форматы кадров технологии Fast Ethernet не отличаются от форматов кадров технологий Ethernet 10 Мбит/с.
>
> Межкадровый интервал равен 0,96 мкс, а битовый интервал — 10 нс. Все временны́е параметры алгоритма доступа (интервал отсрочки, время передачи кадра минимальной длины и т. п.), измеренные в битовых интервалах, остались прежними.
>
> Признаком свободного состояния среды является передача по ней символа простоя источника — соответствующего избыточного кода (а не отсутствие сигналов, как в стандартах Ethernet 10 Мбит/с).

Версия 100Base-T4 носила промежуточный характер, так как позволяла повысить скорость классического варианта Ethernet в 10 раз, не меняя кабельную систему здания. Так как большинство предприятий и организаций достаточно быстро заменили кабели категории 3 кабелями категории 5, то необходимость в версии 100Base-T4 отпала, после чего оборудование с таким портами перестало выпускаться, в связи с чем далее рассмотрены детали только спецификаций 100Base-FX и 100Base-TX.

Спецификация 100Base-FX определяет работу протокола Fast Ethernet по многомодовому оптоволокну в полудуплексном и дуплексном режимах. Подуровень кодирования 100Base-FX преобразует байты, получаемые от уровня MAC, в избыточные коды 4B/5B (см. главу 7). В качестве физического метода кодирования используется код NRZ («свет-темнота»). Существование запрещенных комбинаций символов позволяет отбраковывать ошибочные символы, что повышает устойчивость работы сетей 100Base-FX/TX. Так, в Fast Ethernet признаком того, что среда свободна, стала повторяющаяся передача одного из запрещенных для кодирования пользовательских данных символа, а именно символа

простоя источника *Idle* (11111). Такой способ позволяет приемнику всегда находиться в синхронизме с передатчиком. Для отделения кадра Ethernet от символов простоя источника используется комбинация символов начального ограничителя кадра — пара символов *J* (11000) и *K* (10001) кода 4B/5B, а после завершения кадра перед первым символом простоя источника вставляется символ *T* (рис. 10.17).



Рис. 10.17. Непрерывный поток данных спецификаций 100Base-FX/TX

После преобразования 4-битных порций кодов MAC в 5-битные порции физического уровня их необходимо представить в виде оптических или электрических сигналов в кабеле, соединяющем узлы сети.

В спецификации 100Base-TX в качестве среды передачи данных используется витая пара UTP категории 5. Как и в спецификации 100Base-FX, здесь применяется избыточный код 4B/5B, а для физического кодирования — код MLT-3 с тремя состояниями электрического сигнала (один из вариантов кода NRZ). Основным отличием от спецификации 100Base-FX является наличие схемы автопереговоров для выбора режима работы порта.

Gigabit Ethernet

Уже вскоре после появления на рынке продуктов Fast Ethernet сетевые интеграторы и администраторы при построении корпоративных сетей почувствовали определенные ограничения. Во многих случаях серверы, подключенные по 100-мегабитному каналу, перегружали магистрали сетей, также работающие на скорости 100 Мбит/с — магистрали FDDI и Fast Ethernet. Ощущалась потребность в следующем уровне иерархии скоростей. Стандарт Ethernet с битовой скоростью 1000 Мбит/с, получивший название Gigabit Ethernet, был принят в 1998 году.

Проблемы совместимости

Основная идея разработчиков стандарта Gigabit Ethernet состояла в максимальном сохранении идей классической технологии Ethernet при достижении битовой скорости в 1000 Мбит/с.

В результате дебатов были приняты следующие решения:

- сохраняются все форматы кадров Ethernet;
- по-прежнему существует полудуплексная версия протокола, поддерживающая метод доступа CSMA/CD;
- поддерживаются все основные виды кабелей, используемых в Ethernet и Fast Ethernet, в том числе волоконно-оптический кабель, витая пара категории 5, экранированная витая пара.

Несмотря на то что в Gigabit Ethernet не стали встраивать новые функции, поддержание даже достаточно простых функций классического стандарта Ethernet на скорости 1 Гбит/с потребовало решения нескольких сложных задач.

- *Обеспечение приемлемого диаметра сети для работы на разделяемой среде.* В связи с ограничениями, накладываемыми методом CSMA/CD на длину кабеля, версия Gigabit Ethernet для разделяемой среды допускала бы длину сегмента всего в 25 м при сохранении размера кадров и всех параметров метода CSMA/CD неизменными. Так как существует большое количество применений, требующих диаметра сети 100 м, необходимо было каким-то образом решить эту задачу за счет минимальных изменений в технологии Fast Ethernet.
- *Достижение битовой скорости 1000 Мбит/с на оптическом кабеле.* Технология Fibre Channel, физический уровень которой был взят за основу оптоволоконной версии Gigabit Ethernet, обеспечивала скорость передачи данных всего в 800 Мбит/с.
- *Использование в качестве кабеля витой пары.* Такая задача на первый взгляд кажется неразрешимой — ведь даже для 100-мегабитных протоколов требуются достаточно сложные методы кодирования, чтобы уложить спектр сигнала в полосу пропускания кабеля.

Для решения этих задач разработчикам технологии Gigabit Ethernet пришлось внести изменения не только в физический уровень, как это было в случае Fast Ethernet, но и в уровень MAC.

Средства обеспечения диаметра сети в 200 м на разделяемой среде

Для расширения максимального диаметра сети Gigabit Ethernet до 200 м в полудуплексном режиме разработчики технологии предприняли достаточно естественные меры, в основе которых лежало известное соотношение времени передачи кадра минимальной длины и времени оборота (PDV). Минимальный размер кадра был увеличен (без учета преамбулы) с 64 до 512 байт, или до 4096 бит. Соответственно, время оборота увеличилось до 4095 битовых интервалов, что при использовании одного повторителя сделало допустимым диаметр сети около 200 м.

Для увеличения длины кадра до величины, требуемой в новой технологии, сетевой адаптер должен дополнить поле данных до длины 448 байт так называемым **расширением**, представляющим собой поле, заполненное нулями. Формально минимальный размер кадра не изменился (64 байт или 512 бит), и объясняется это тем, что поле расширения помещается после поля контрольной суммы кадра (FCS). Соответственно, значение этого поля не включается в контрольную сумму и не учитывается при указании длины поля данных в поле длины. Поле расширения является просто расширением сигнала несущей частоты, необходимым для корректного обнаружения коллизий.

Для сокращения накладных расходов в случае использования слишком длинных кадров при передаче коротких квитанций разработчики стандарта разрешили конечным узлам *передавать несколько кадров подряд без возвращения среды* другим станциям. Такой режим получил название **режима пульсаций**. Станция может передать подряд несколько кадров с общей длиной не более 65 536 бит, или 8192 байт. При передаче нескольких небольших кадров станции можно не дополнять первый кадр до размера в 512 байт за счет поля расширения, а передавать несколько кадров подряд до исчерпания предела

в 8192 байт (в этот предел входят все байты кадра, в том числе преамбула, заголовок, данные и контрольная сумма). Предел 8192 байт называется **длиной пульсации**. Если предел длины пульсации достигается в середине кадра, то кадр разрешается передать до конца. Увеличение «совмещенного» кадра до 8192 байт несколько задерживает доступ к разделяемой среде других станций, но при скорости 1000 Мбит/с эта задержка не столь существенна.

Вариант Gigabit Ethernet на разделяемой среде сегодня не имеет практического значения, но мы привели его описание в качестве примера интересного инженерного решения.

Спецификации физической среды стандарта Gigabit Ethernet

В стандарте Gigabit Ethernet определены следующие типы физической среды:

- одномодовый волоконно-оптический кабель;
- многомодовый волоконно-оптический кабель 62,5/125;
- многомодовый волоконно-оптический кабель 50/125;
- экранированный сбалансированный медный кабель.

Спецификация 1000Base-LX (L означает Long Wavelength — длинная волна) работает с оптическим сигналом в окне 1300 нм: при использовании одномодового волокна длина кабеля может достигать 5 км, а многомодового — 550 м. Спецификации 1000Base-SX (S означает Short Wavelength — короткая длина волны) использует сигнал в окне 850 нм и работает только на многомодовом волокне длиной до 220–550 м в зависимости от толщины сердечника волокна.

В спецификациях 1000Base-SX и 1000Base-LX подуровень кодирования преобразует байты уровня MAC в коды 8B/10B (а не **4**B/5B, как в стандарте Fast Ethernet).

Gigabit Ethernet на витой паре категории 5

Как известно, каждая пара кабеля категории 5 имеет гарантированную полосу пропускания до 100 МГц. Организовать передачу по такому кабелю данных со скоростью 1000 Мбит/с очень трудно, так как применяемые в локальных сетях методы кодирования имеют на такой тактовой частоте спектр с шириной, намного превышающей 100 МГц. В качестве решения было предложено организовать параллельную передачу данных одновременно по всем четырем парам кабеля. Это сразу снизило скорость передачи данных по каждой паре до 250 Мбит/с. Однако и для такой скорости необходимо было придумать метод кодирования со спектром, не превышающим 100 МГц. Усложняло задачу и то обстоятельство, что стандарт Gigabit Ethernet должен поддерживать не только полудуплексный, но и дуплексный режим. На первый взгляд кажется, что одновременное использование четырех пар лишает сеть возможности работы в дуплексном режиме, так как не остается свободных пар для одновременной передачи данных в двух направлениях — от узла и к узлу. Тем не менее группа 802.3ab нашла решения обеих проблем.

Для физического кодирования данных был применен код PAM5 с пятью уровнями потенциала: –2, –1, 0, +1, +2. В этом случае за один такт по одной паре передается 2,322 бит информации ($\log_2 5$). Следовательно, для достижения скорости 250 Мбит/с тактовую частоту 250 МГц можно уменьшить в 2,322 раза. Разработчики стандарта решили использовать

несколько более высокую частоту, а именно 125 МГц. При этой тактовой частоте код PAM5 имеет спектр у́же, чем 100 МГц, то есть он может быть передан без искажений по кабелю категории 5. В каждом такте передается не 2,322 × 4 = 9,288 бит информации, а 8. Это и дает искомую суммарную скорость 1000 Мбит/с. Передача ровно восьми битов в каждом такте достигается за счет того, что подуровень кодирования PCS преобразует байты, получаемые от уровня MAC через интерфейс GMII, в логический код 4D-PAM5, который состоит из 4 символов, каждый их которых имеет 5 состояний (5 состояний символа соответствует 5 состояниям физического кода PAM5). Отметим, что при кодировании информации используются не все 625 ($5^4 = 625$) комбинаций кода PAM5, а только 256 ($2^8 = 256$) — оставшиеся приемник задействует для контроля принимаемой информации и выделения правильных комбинаций на фоне шума.

Для организации дуплексного режима разработчики спецификации 802.3ab применили технику выделения принимаемого сигнала из суммарного. Два передатчика работают навстречу друг другу по каждой из четырех пар в одном и том же диапазоне частот. Для отделения принимаемого сигнала от собственного приемник вычитает из результирующего сигнала известный ему свой сигнал. Естественно, это непростая операция — для ее выполнения используются специальные процессоры цифровой обработки сигнала (Digital Signal Processor, DSP).

Существуют и другие спецификации физической среды для Gigabit Ethernet.

10G Ethernet

Стандарт **10G Ethernet** определяет только дуплексный режим работы, поэтому он используется исключительно в *коммутируемых* локальных сетях Этот стандарт может работать только на волоконно-оптическом кабеле.

Формально этот стандарт имеет обозначение **IEEE 802.3ae** и является дополнением к основному тексту стандарта 802.3. Формат кадра остался неизменным, расширение кадра, введенное в стандарте Gigabit Ethernet, не используется, так как нет необходимости обеспечивать распознавание коллизий.

Стандарт 802.3ae, принятый в 2002 году, описывает несколько новых спецификаций физического уровня, взаимодействующих с уровнем MAC через новый интерфейс **XGMII** (eXtended Gigabit Medium Independent Interface — расширенный интерфейс независимого доступа к гигабитной среде). Интерфейс XGMII предусматривает параллельный обмен четырьмя байтами, образующими четыре потока данных. Как видим, идея распараллеливания потоков данных, хорошо зарекомендовавшая себя в предыдущих версиях Ethernet как средство снижения требований к полосе пропускания отдельного канала, здесь нашла отражение в структуре интерфейса между уровнем MAC и физическим уровнем. Наличие четырех независимых потоков на входе физического уровня упрощает организацию параллельных потоков данных в приемопередатчиках Ethernet. Стандарт 10G Ethernet определяет три группы физических интерфейсов:

- 10GBase-X;
- 10GBase-R4;
- 10GBase-W.

Они отличаются способом логического кодирования данных: в варианте 10Base-X применяется код 8B/10B, в остальных двух — код 64B/66B. Для передачи данных все они используют оптическую среду.

100G и 40G Ethernet

Скорость в 10 Гбит/с является довольно высокой скоростью передачи данных, достаточной для многих приложений, например, для телевидения высокого разрешения, но и она по прошествии нескольких лет перестала удовлетворять потребности постоянно растущего трафика интернета и новых приложений. Стандарт Ethernet 40G/100G 802.3ba обобщает подход, использованный в технологии 10G Ethernet, а именно — распараллеливание общего потока данных от уровня MAC на несколько потоков, что позволяет снизить битовую скорость каждого из параллельных потоков, тем самым упрощая реализацию высокоскоростного приемопередатчика.

В стандарте 802.3ba распараллеливание применяется на двух этапах передачи данных от уровня MAC к подуровню зависимости от физической среды PMD. Сначала уровень согласования распараллеливает общий последовательный поток данных, поступающий от уровня MAC, на 8 потоков, параллельно поступающих на подуровень физического кодирования PCS через интерфейс **XLGMII** (для скорости 40 Гбит/с, буквы XL означают римское число 40) или интерфейс **CGMII** (для скорости 100 Гбит/с, от C — римское число 100). Подуровень PCS выполняет кодирование данных, поступающих по 8 потокам, в соответствии с кодировкой 64B/66B (общая для всех вариантов физического интерфейса), направляя их четырьмя (для скорости 40 Гбит/с) или двадцатью (для скорости 100 Гбит/с) потоками на подуровни PMA и PMD, реализуемые, как правило, отдельным модулем — приемопередатчиком (трансивером). В подуровнях PMA/PMD потоки могут группироваться в один или несколько каналов, передаваемых отдельными волнами (если применяется мультиплексирование WDM) или отдельными медными проводниками. Выбор двадцати потоков не случаен — он обеспечивает высокую гибкость для спецификаций физической среды, так как дает возможность сформировать 1, 2, 4, 5, 10 или 20 независимых физических каналов. Рисунок 10.18 иллюстрирует работу подуровня PCS по распараллеливанию данных по потокам.

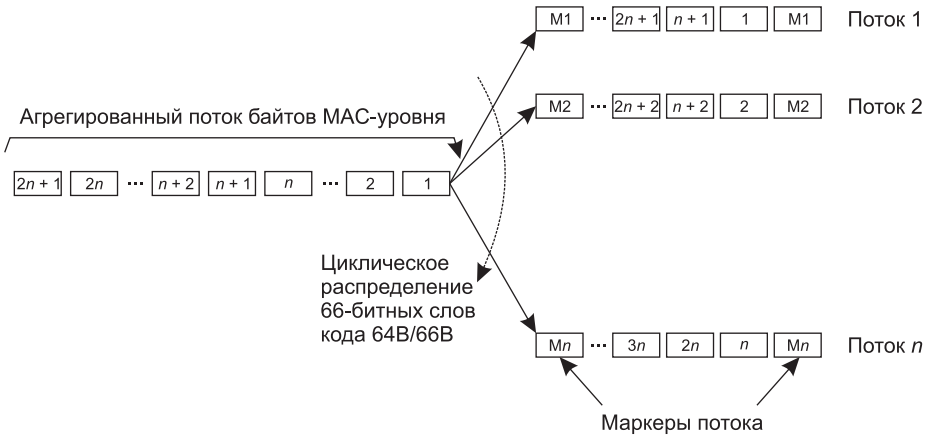


Рис. 10.18. Распределение данных подуровнем PCS по потокам

Шестидесятишестибитные слова кода 64B/66B циклически распределяются между потоками данных.

Для 100 Gigabit Ethernet рабочей группой IEEE 802.3 разработаны различные стандарты физического уровня, работающие на расстоянии 100–150 м на многомодовом кабеле и до 40 км на одномодовом. Почти все варианты физического уровня 100GE обеспечивают необходимую суммарную битовую скорость за счет использования нескольких параллельных потоков данных — четырех или десяти. Эти параллельные потоки образуются либо отдельными проводниками, например парами оптических волокон, либо отдельными волнами технологии CWDM. Во всех вариантах физического уровня 100G Ethernet использует модуляцию NRZ с двумя состояниями сигнала и скремблирование для устранения эффекта длинной последовательности нулей в коде 64B/66B. При применении 10 потоков скорость отдельного потока составляет 10 Гбит/с, а при применении четырех — 25 Гбит/с.

Примерами стандартов100G Ethernet являются:

- стандарт 100GBase-LR4, который передает 4 потока данных по четырем различным оптическим волнам;
- стандарт 100GBase-SR4, который передает 4 потока данных по четырем различным парам оптических волокон.

Первое поколение 200G и 400G Ethernet

Довольно скоро после принятия стандарта 40G Ethernet сетевые адаптеры этой скорости стали активно применяться в серверах центров обработки данных, а это означало, что нужно было пропорционально увеличить скорость интерфейсов магистральных маршрутизаторов интернета, чтобы магистрали продолжали справляться со все возрастающим объемом трафика. Для решения этой проблемы был разработан стандарт IEEE 802.bs, (конец 2017 года), который определил спецификации нескольких вариантов физического уровня для скоростей 200 и 400 Гбит/с.

Стандарты 200G и 400G Ethernet базируются на подходе, примененном в стандарте 100G Ethernet, — распараллеливании данных на несколько потоков, передаваемых по отдельным физическим проводникам: электрическим или оптическим. Отличие в том, что теперь скорость каждого потока повышена до 50 Гбит/с. Для повышения битовой скорости потока применено два приема: кодирование PAM4 с четырьмя состояниями электрического (оптического) сигнала либо добавление кодов FEC к потоку битов данных.

Переход на метод кодирования с большим числом состояний был очень желателен для достижения скорости 400 Гбит/с, так как при кодировании NRZ максимально достижимая битовая скорость — около 25 Гбит/с — была во время разработки стандарта 400 G (2017–2018 гг.) ограничена пределом: тактовой скоростью в 25–30 Гбод электронных схем. Применение кодирования NRZ привело бы к необходимости использования 16 параллельных «дорожек» передачи информации (оптических волокон или волн) со скоростью по 25 Гбит/с для одного направления скорости 400 Гбит/с, то есть к необходимости производства кабелей локальных сетей с 32 оптическими волокнами. Такой вариант кабеля возможен, но это не очень масштабируемое решение, если считать, что требования к скорости будут повышаться и в дальнейшем. Переход на новый метод кодирования с бóльшим количеством состояний сигнала был признан более перспективным. Но, как мы знаем, при повышении числа состояний сигнала увеличивается число отношений «сигнал — шум»

в линии связи, а следовательно, и частота битовых ошибок в принимаемых сигналах. Поэтому решение применить коды FEC было вынужденным, иначе кодирование PAM4 не позволило бы осуществить надежную передачу информации с нужной скоростью. Стандартная контрольная сумма CRC-32 кадра Ethernet не в состоянии решить эту задачу.

На рис. 10.19, *а* показаны подуровни физического уровня стандарта 400G Ethernet (мы сначала опишем детали реализации этой скорости, а потом покажем, каким образом решения, найденные для 400G, были использованы для версий 200G и 50G).

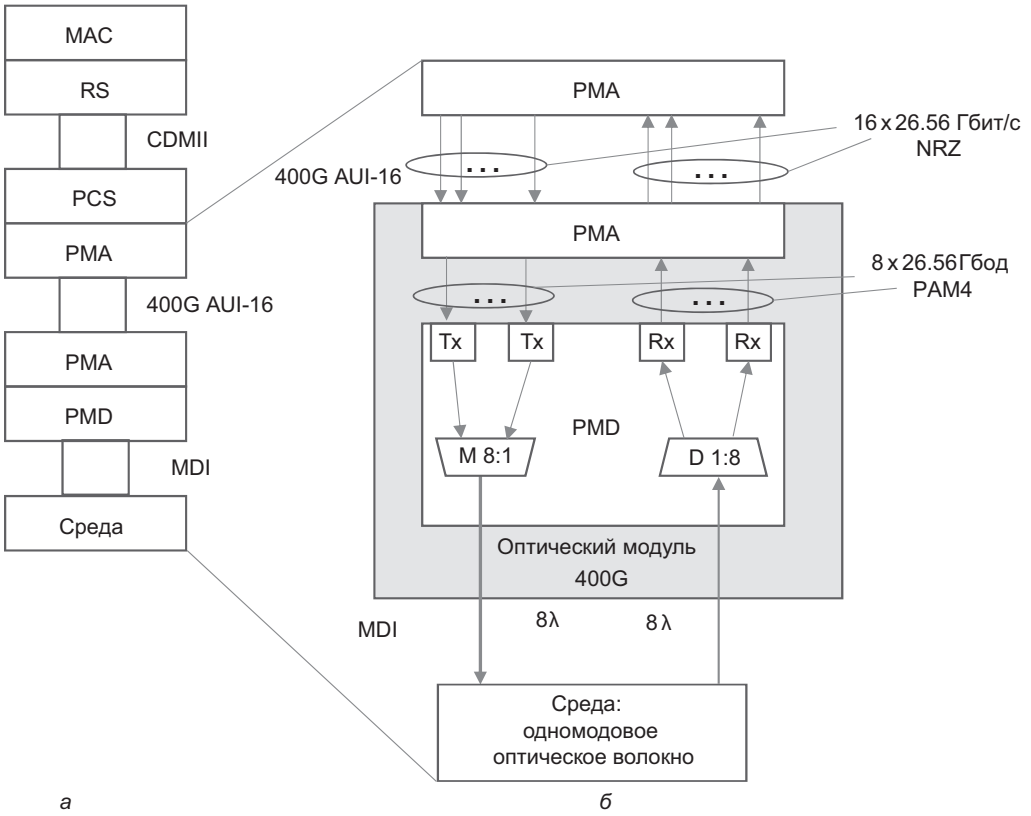


Рис. 10.19. Реализация физического уровня Ethernet 400G

Подуровень физического кодирования PCS усложнился по сравнению с предыдущими версиями Ethernet, так как именно он добавляет коды FEC к потоку данных, получаемому от уровня MAC через подуровень согласования RS через интерфейс CDMII (CD — 400 в римской нотации). В качестве кодов FEC рекомендуется использовать коды Рида — Соломона RS (544, 514, 10). Это означает, что к каждым 514 словам исходного кода добавляется 30 контрольных слов, так что результирующий блок данных имеет длину 544 слова. Последний параметр в обозначении кода говорит о том, что слово имеет длину в 10 бит. При генерировании кодов FEC данные, получаемые от уровня MAC, рассматриваются как поток битов, разбиение их на поля кадра Ethernet игнорируется.

В этом отношении реализация механизма FEC в Ethernet 400G похожа на его реализацию в технологии OTN, в которой коды FEC также периодически вставляются в поток битов для каждого блока битов определенной длины. Уровень PCS узла-приемника использует коды FEC для контроля и исправления битовых ошибок, а поле этого узла-приемника удаляет их из потока битов, так что уровень MAC узла-приемника получает кадр в стандартной форме. Коды FEC работают на участке между непосредственно соединенными узлами. После вычисления и добавления кодов FEC подуровень скремблирует данные и распределяет их на 16 потоков, передаваемых подуровню физического присоединения PMA.

Подуровень PMA обычно разбивается на две части, как это показано на рис. 10.19, *б*, одна из которых принадлежит порту Ethernet 400G коммутатора (маршрутизатора), а вторая — оптическому модулю 400G, устанавливаемому в порт (называемому также трансивером, см. материал «Конструктивное исполнение коммутаторов» на сайте книги). Обмен данными между этими частями происходит по электрическому интерфейсу 400G AUI-16, который состоит из 16 электрических контактов, по которым передаются 16 потоков в коде NRZ. Скорость передачи данных в каждом потоке составляет 25,56 Гбит/с, что в сумме дает 425 Гбит/с — эта скорость превышает требуемые 400 Гбит/с из-за необходимости передавать коды FEC. Заметим, что скорость изменения сигнала в потоке равна 25,56 Гбод, так как она всегда совпадает со скоростью передачи данных при кодировании NRZ с двумя состояниями сигнала. Подуровень PMA оптического модуля принимает 16 потоков NRZ и объединяет их в 8 потоков, перекодируя данные в код PAM4. При этом скорость изменения сигнала в каждом потоке не изменяется — она остается равной 25,56 Гбод (что хорошо для электронных компонент оптического модуля), но скорость передачи данных увеличивается вдвое и теперь равна 53,12 Гбит/с.

На рис. 10.19, *б* показан один из вариантов использования потоков 53,12 Гбит/с (их часто для краткости называют потоками 50 Гбит/с) оптическим модулем для передачи их по двум волокнам одномодового оптоволоконного кабеля. Передатчики Tx преобразуют электрические сигналы в оптические, при этом для каждого потока используется отдельная волна из диапазона CWDM (диапазон состоит из 16 волн длиной от 1270 нм до 1610 нм с шагом 20 нм). Эти сигналы поступают на мультиплексор M, выдающий составной бесцветный сигнал на передающее волокно оптоволоконного кабеля. При поступлении составного сигнала из приемного волокна кабеля он демультиплексируется на отдельные волны, оптические сигналы которых затем преобразуются в электрические приемниками Rx.

Описанный ранее физический стандарт 400G Ethernet, использующий передачу на двух одномодовых волокнах оптического кабеля определенной длины, носит название 400GBase-FR8, 400GBase-SR8 или 400GBase-ER8 в зависимости от максимальной длины кабеля (2 км, 10 км или 40 км соответственно). В случае многомодового оптического кабеля применяется параллелизм волокон кабеля, а не волн. Так, стандарт 400GBase-SR8 работает на кабеле, состоящем из 8 волокон для каждого направления.

Поток данных скорости 53,12 Гбит/с с кодировкой PAM4 стал основным элементом не только стандарта 400G, но и стандартов 200G и 50G. Отличие стандартов 200GBase-FR4, 200GBase-SR4 и 200GBase-ER4 от своих более скоростных собратьев состоит только в том, что в них применяются не восемь, а четыре потока 50 Гбит/с, передаваемых четырьмя волнами в каждом направлении. Соответственно, в стандартах 50GBase-SR/LR применяются один поток 50 Гбит/с и одна длина волны в каждом направлении.

Новые поколения: 400G (4 линии), 800G и 1,6T Ethernet

Прогресс в области микроэлектроники обусловил появление новых поколений высокоскоростного Ethernet. В этих технологиях скорость модуляции сигналов на одной внутренней линии сетевого адаптера возрастает до 50 Гбод/100 Гбод в отличие от скорости 25 Гбод стандартов первого поколения. За счет этого достигается либо повышение скорости передачи данных до 1,6 Тбит/с, либо, при сохранении той же скорости 400 Гбит/с, удается уменьшить число необходимых для передачи оптических волокон/волн.

Сначала (2019–2022 гг.) применение новых поколений цифровых процессоров, способных компенсировать искажения сигналов на сверхвысоких частотах, позволило повысить скорость модуляции кода PAM4 на отдельной линии до 53–56 Гбод. Это привело к появлению поколения стандартов Ethernet со скоростью одной линии 100 Гбит/с:

- **100GBase-FR1** — в отличие от стандартов 100G Ethernet первого поколения в этом стандарте передача данных идет внутри сетевого адаптера по одной линии, а между сетевыми адаптерами — по паре одномодового оптического волокна на расстояние до 2 км.
- **400GBase-FR4** — используется передача внутри сетевого адаптера по четырем отдельным линиям на скорости 100 Гбит/с, а между сетевыми адаптерами — по четырем отдельным световым волнам пары одномодовых оптоволокон. Таким образом, на смену технологии 400G первого поколения с 8 проводниками между сетевыми адаптерами пришло новое поколение технологии 400G с 4 проводниками.

Дальнейший прогресс в области цифровых методов обработки сигналов привел к появлению сетевых адаптеров Ethernet с частотой модуляции 100–112 Гбод, что при модуляции PAM-4 дало возможность создать сетевые адаптеры со скоростью одной линии 200 Гбит/с. Такая скорость одной линии позволила создать стандарты Ethernet с битовой скоростью 800 Гбит/с. Например, стандарт 800GBase-FR передает свои данные по отдельным четырем линиям со скоростью 200 Гбит/с каждая внутри сетевого адаптера, и по 4-м световым волнам одной пары одномодового оптоволоконного кабеля между сетевыми адаптерами на расстоянии до двух километров. В то же время, стандарт 800G также использует и внутренние линии со скоростью 100 Гбит/с, например, стандарт 800GBase-SR8 передает данные по 8 парам многомодового оптоволоконного кабеля на расстояние до 100 м. Выбор между использованием линии 100 Гбит/с и 200 Гбит/с для конкретного варианта происходит на основании требований к стоимости сетевого адаптера и максимального расстояния передачи.

Несмотря на возможность использования линий со скоростью 200 Гбит/с, стандарты 400G Ethernet по-прежнему работают на линиях 100 Гбит/с, так как в период их стандартизации линии 200 Гбит/с не были доступны. Возможно, эта ситуация изменится в будущем.

Как мы знаем, в сетях OTN/DWDM применяются такие схемы модуляции, как 16-QAM или 64-QAM, которые позволяют передавать 4 или 6 бит информации за один такт (бод) работы модулятора. Однако в описанных нами высокоскоростных версиях Ethernet они не применяются. Причина состоит в том, что такие эффективные методы модуляции требуют применения когерентных лазеров и другого дорогого оборудования, что намного бы увеличило стоимость сетевых адаптеров сетей Ethernet. Тем не менее консорциум OIF (Optical Internetworking Forum) стандартизовал физические уровни Ethernet с такими характеристиками в своих спецификациях 400GBase-ZX и 400Base-ZX+. Эти специфика-

ции созданы для оборудования с интерфейсами Ethernet, работающими непосредственно через соединения DWDM, без инкапсуляции кадров Ethernet в кадры технологии OTN. Спецификация 400GBase-ZX применяет 8 линий 50 Гбит/с с модуляцией PAM-4 внутри сетевого адаптера, а на выходе эти сигналы объединяются в один оптический сигнал, использующий когерентную оптику с модуляцией 16-QAM для расстояний до 500 км, и с модуляцией QPSK для расстояний до 2000 км. Такие интерфейсы не требуют инкапсуляции кадров Ethernet в кадры OTN, а это означает меньшие накладные расходы и меньшие задержки при передаче данных.

Следующей ступенью в иерархии скоростей Ethernet будет скорость 1,6 Тбит/с. Во время написания книги эта версия Ethernet еще не была стандартизована, но ее основные параметры уже были известны. Она будет использовать 8 линий со скоростью 200 Гбит/с и модуляцией PAM-4.

(S) *Конструктивное исполнение ком.мутаторов*

ГЛАВА 11 Отказоустойчивые и виртуальные локальные сети

Алгоритм покрывающего дерева

В коммутируемых локальных сетях проблема обеспечения надежности сети имеет свою специфику: базовый протокол прозрачного моста корректно работает только в сети с *древовидной топологией*, в которой между любыми двумя узлами сети существует единственный маршрут. Тем не менее очевидно, что для надежной работы сети необходимо наличие альтернативных маршрутов между узлами, которые можно использовать при отказе основного маршрута. Наиболее простое решение этой проблемы — построение сети с альтернативными маршрутами, ручное нахождение связной древовидной топологии и ручное блокирование (то есть перевод в административное состояние «отключен») всех портов, которые не входят в найденную топологию. В случае отказа сети этот процесс должен повторяться — опять же в ручном режиме. Понятно, что надежность сети в этом случае оказывается не очень высокой, так как время пребывания ее в неработоспособном состоянии будет исчисляться минутами: нужно обнаружить отказ, локализовать его (то есть не только зафиксировать факт, что в сети что-то перестало работать, но и понять, какая именно связь пострадала и требует обхода), затем найти новый работоспособный вариант топологии сети (если он, конечно, существует) и сконфигурировать его.

Для автоматического выполнения перечисленных действий, то есть нахождения и конфигурирования активной древовидной топологии, мониторинга состояния ее связей и перехода к новой древовидной топологии при обнаружении отказа связи в коммутируемых локальных сетях используются **алгоритм покрывающего дерева** (Spanning Tree Algorithm, STA) и реализующий его **протокол покрывающего дерева** (Spanning Tree Protocol, STP).

Алгоритм покрывающего дерева, разработанный в 1983 году, был признан IEEE удачным решением и включен в ту же спецификацию 802.1D, в которой описывается и алгоритм прозрачного моста. Фактически протокол STP является специфической упрощенной версией протокола маршрутизации. Упрощение — в том, что кадры направляются по активному маршруту независимо от их адреса назначения, в то время как в протоколах маршрутизации активный маршрут выбирается для каждого адреса индивидуально. Сегодня протокол STP широко применяется в наиболее массовых устройствах современных локальных сетей — коммутаторах. Версия протокола STP, получившая название RSTP (Rapid STP, то есть быстрый протокол покрывающего дерева), затрачивает на поиск новой топологии несколько секунд.

Протокол STP

Протокол STP формализует сеть (рис. 11.1, *а*) в виде графа (рис. 11.1, *б*), вершинами которого являются коммутаторы и сегменты сети. **Сегмент** — это связная часть сети, не содержащая коммутаторов (маршрутизаторов). Сегмент может быть разделяемым (во времена создания алгоритма STA это был единственный тип сегмента) и включать устройства физического уровня — повторители/концентраторы, существование которых коммутатор, будучи устройством канального уровня, «не замечает».

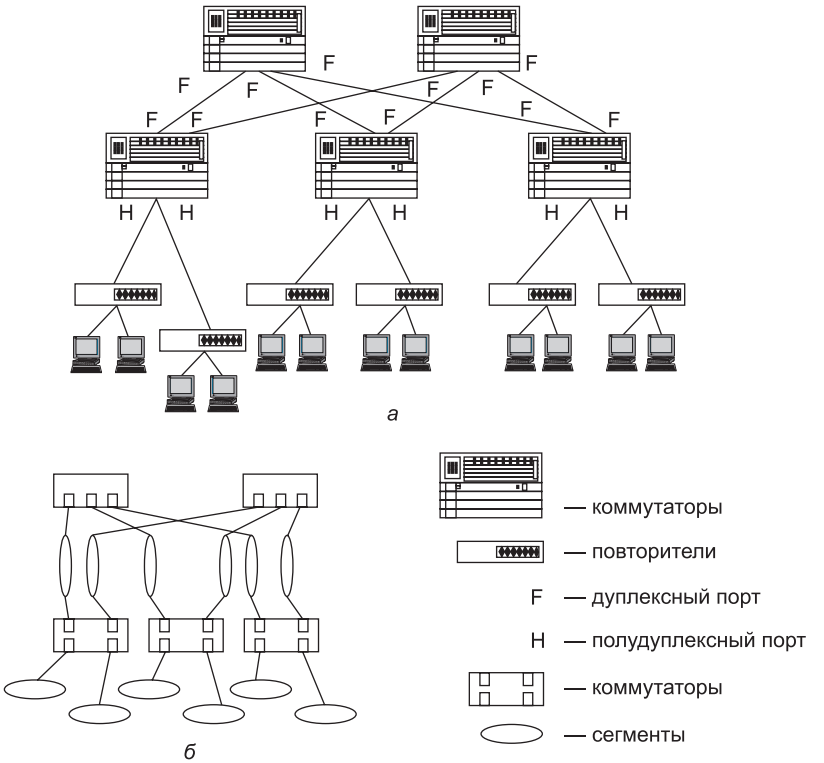


Рис. 11.1. Формализованное представление сети в соответствии с алгоритмом STA

Сегмент может представлять собой и двухточечный канал. В коммутируемых локальных сетях, применяемых сегодня, это единственный тип сегмента.

Протокол покрывающего дерева обеспечивает построение древовидной топологии связей с единственным путем минимальной длины от каждого коммутатора и от каждого сегмента до некоторого выделенного **корневого коммутатора** — корня дерева. *Единственность* пути гарантирует отсутствие петель, а *минимальность* расстояния — рациональность маршрутов следования трафика от периферии сети к ее магистрали, роль которой исполняет корневой коммутатор.

В качестве расстояния в STA используется **метрика** — традиционная для протоколов маршрутизации величина, обратно пропорциональная пропускной способности сегмента.

Также в STA метрика определяется как *условное время передачи бита сегментом*. В текущей версии стандарта 802.1D-2004 используются такие значения метрик, которые расширяют диапазон скоростей сегментов до 10 Тбит/с (то есть с большим запасом относительно сегодняшнего уровня максимальной для Ethernet скорости в 100 Гбит/с), давая такому сегменту значение 2; соответственно, сегмент 100 Гбит/с получает значение 200, 10 Гбит/с — 2000, 1 Гбит/с — 20 000, 100 Мбит/с — 200 000, а 10 Мбит/с — 2 000 000.

Протокольными единицами данных моста (Bridge Protocol Data Unit, **BPDU**) называются специальные пакеты, которыми периодически обмениваются коммутаторы для автоматического определения конфигурации дерева. Пакеты BPDU переносят данные об идентификаторах коммутаторов и портов, а также о расстоянии до корневого коммутатора. Существуют два типа сообщений, которые переносят пакеты BPDU: конфигурационные сообщения, называемые также сообщениями HELLO (с интервалом 2 с), и сообщения с уведомлениями об изменении конфигурации. Для доставки BPDU используется групповой адрес 01:80:C2:00:00:00, позволяющий организовать эффективный обмен данными.

На рисунке пути трафика сигнальных протоколов показаны пунктиром, пути трафика данных — сплошной линией.

Три этапа построения дерева

На рис. 11.2 приведен пример сети стандарта 802.1D-2004, который иллюстрирует работу протокола STP. Мы также будем использовать этот пример в своем описании.

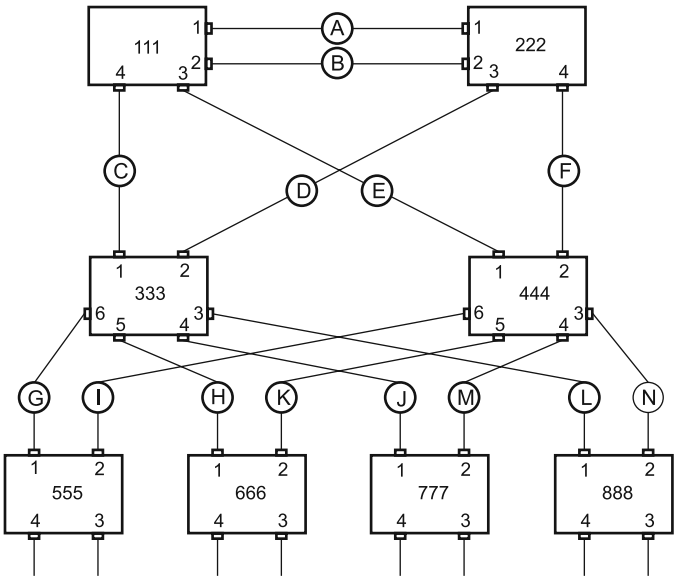


Рис. 11.2. Пример сети, иллюстрирующей работу STP

В этом примере сеть построена на восьми коммутаторах, имеющих идентификаторы со значениями от 111 до 888. В стандарте в качестве идентификатора коммутатора используется его MAC-адрес, к которому добавляются два старших байта, конфигурируемые

вручную, — администратор может использовать их для вмешательства в выбор корневого коммутатора. Для удобства записи на рисунке указаны сокращенные до трех разрядов значения MAC-адресов коммутаторов. Все коммутаторы соединены друг с другом двухточечными связями, которые образуют сегменты *A*–*N*. Порты 3 и 4 коммутаторов с 555 по 888 соединены с конечными узлами сети — компьютерами (на рисунке не показаны). Все связи в сети — связи со скоростью 100 Мбит/с (Fast Ethernet). Алгоритм STA определяет активную конфигурацию сети в три этапа:

1. *Определение корневого коммутатора, от которого строится дерево.* В качестве корневого коммутатора выбирается коммутатор с *наименьшим значением идентификатора.* В исходном состоянии каждый коммутатор, считая себя корневым, генерирует и передает своим соседям сообщения HELLO, в которых помещает свой идентификатор в качестве идентификатора корневого коммутатора. Коммутатор, получив от соседа сообщение HELLO, содержащее идентификатор корневого коммутатора, меньший его собственного, перестает считать себя корневым коммутатором и генерировать свои сообщения HELLO, начиная ретранслировать сообщения HELLO, получаемые от соседей. В нашем примере предполагается, что администратор не стал менять старшие байты коммутаторов, так что у всех коммутаторов они остались равными значению 32 768 (значение, предлагаемое по умолчанию), а корневым коммутатором стал коммутатор с идентификатором 111.
2. *Выбор корневого порта для каждого коммутатора.* Корневым портом коммутатора является тот порт, расстояние от которого до корневого коммутатора является минимальным. Сам корневой коммутатор корневых портов не имеет. Для определения корневого порта каждый коммутатор использует пакеты HELLO, ретранслируемые ему другими коммутаторами. На основании этих пакетов каждый коммутатор определяет минимальные расстояния от всех своих портов до корневого коммутатора и выбирает порт с наименьшим значением в качестве корневого. При равенстве расстояний выбирается порт с наименьшим значением идентификатора порта (это порядковый номер порта в коммутаторе).

   Например, у коммутатора 222 порты 1 и 2 находятся на одинаковом расстоянии до корневого коммутатора 111 — оба эти порта непосредственно связаны через сегменты *A* и *B* с коммутатором 111, а значит, получают пакеты HELLO с метрикой, равной 0. Так как идентификатор порта 1 меньше идентификатора порта 2, то корневым портом коммутатора 222 выбирается порт 1. Аналогично корневым портом коммутатора 555 становится порт 1, а не порт 2. Оба эти порта получают сообщения HELLO, генерируемые корневым коммутатором 111, с наименьшим значением метрики 200 000. Порт 1 получает такие сообщения по маршруту: порт 1 коммутатора 111 — сегмент *C* — порт 1 коммутатора 333 — порт 6 коммутатора 333 — сегмент *G*; соответственно, порт 2 получает их по маршруту: порт 3 коммутатора 111 — сегмент *E* — порт 1 коммутатора 444 — порт 6 коммутатора 444 — сегмент *I*.
3. *Выбор назначенных коммутаторов и портов для каждого сегмента сети.* Назначенным является коммутатор (из числа коммутаторов, непосредственно подключенных к данному сегменту), у которого расстояние до корневого моста является минимальным (точнее, расстояние от корневого порта этого коммутатора до корневого коммутатора). Назначенные порты для сегментов исполняют ту же роль, что и корневые порты для коммутаторов, располагаясь на кратчайшем пути до корневого коммутатора.

Как и при выборе корневого порта, здесь используется распределенная процедура. Каждый коммутатор сегмента прежде всего исключает из рассмотрения свой корневой порт (для

сегмента, к которому он подключен, всегда существует другой коммутатор, расположенный ближе к корню). Для каждого из оставшихся портов выполняется сравнение принятых по ним минимальных расстояний до корня (еще до наращивания на метрику сегмента) с расстоянием до корня корневого порта данного коммутатора. Если все принятые на этом порту расстояния оказываются больше, чем расстояние от собственного корневого порта, то это значит, что для сегмента, к которому подключен порт, кратчайший путь к корневому коммутатору проходит через него, и он становится назначенным. Коммутатор делает все свои порты, для которых такое условие выполняется, назначенными. Когда имеется несколько портов с одинаковым кратчайшим расстоянием до корневого коммутатора, выбирается порт с наименьшим идентификатором.

В нашем примере коммутатор 111 при проверке порта 1 обнаруживает, что через этот порт принимаются пакеты с минимальным расстоянием 200 000 (пакеты от порта 1 коммутатора 222, который ретранслирует через все свои порты сообщения HELLO, полученные от коммутатора 111, но с измененной метрикой, в частности, передает их и коммутатору 111). Так как коммутатор 111 является корневым, его расстояние до корневого коммутатора равно нулю, то есть меньше, чем у получаемых через порт 1 сообщений. Поэтому коммутатор 1 объявляет свой порт 1 назначенным для сегмента *A*. Коммутатор 222 не может объявить свой порт 1 назначенным для сегмента *A*, так как через него он получает сообщения с минимальной метрикой 0, а у его корневого порта метрика равна 200 000. На выполнение всех трех этапов коммутаторам сети отводится по умолчанию 15 секунд. Эта стадия работы портов называется стадией прослушивания (listening) — порты слушают только сообщения BPDU и не передают пользовательских кадров. Считается, что порты находятся в заблокированном состоянии, которое относится только к пользовательским кадрам, в то время как кадры BPDU обрабатываются. Предполагается, что в стадии прослушивания каждый коммутатор получает столько пакетов HELLO, сколько требуется для определения состояния своих портов. Все остальные порты, кроме корневых и назначенных, каждым коммутатором блокируются и не могут передавать пользовательские кадры. Математически доказано, что при таком выборе активных портов из сети исключаются петли, а оставшиеся связи образуют *покрывающее дерево* (если оно вообще может быть построено при существующих связях в сети).

Результат работы протокола STP для нашего примера показан на рис. 11.3.

На рисунке корневые порты коммутаторов отмечены символом *R*: назначенные порты закрашены, а заблокированные зачеркнуты.

После построения покрывающего дерева коммутатор начинает принимать (но не продвигать) пакеты данных и на основе их адресов источника строить таблицу продвижения. Это обычный режим обучения прозрачного моста, который ранее нельзя было активизировать, так как порт не был уверен в том, что он останется корневым или назначенным и будет передавать пакеты данных.

В процессе нормальной работы корневой коммутатор продолжает генерировать пакеты HELLO, а остальные коммутаторы получают их через свои корневые порты и ретранслируют через назначенные порты. Если по истечении максимального времени жизни сообщения (по умолчанию — 10 интервалов HELLO, то есть 20 секунд) корневой порт любого коммутатора сети не получает служебный пакет HELLO, он инициализирует новую процедуру построения покрывающего дерева — так обеспечивается отказоустойчивость локальной сети на коммутаторах. При этом на все порты генерируется и передается пакет HELLO, в котором коммутатор указывает себя в качестве корневого. Аналогично ведут

себя и другие коммутаторы сети, у которых сработал таймер истечения максимального времени жизни сообщения, в результате чего выбирается новая активная конфигурация.

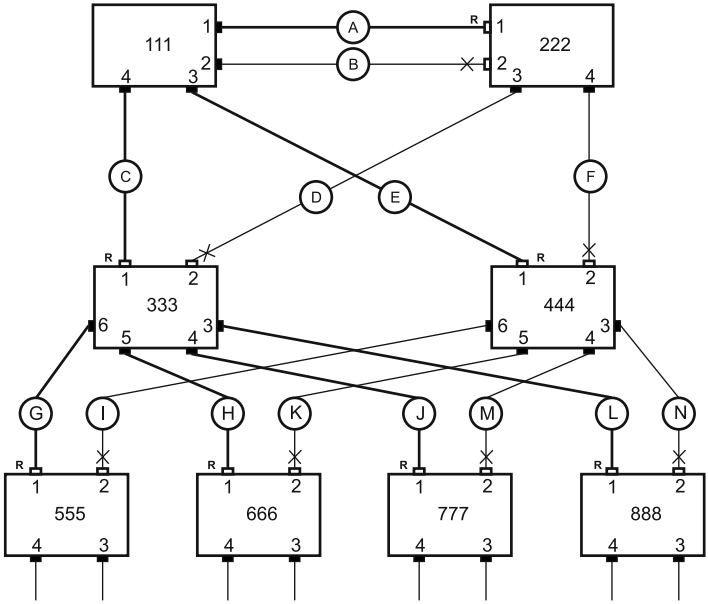


Рис. 11.3. Активная топология, найденная по протоколу STP

Версия RSTP

Основным недостатком протокола STP является его «медлительность»: в сетях с большим количеством коммутаторов время определения новой активной конфигурации может оказаться слишком большим. Если в сети используются заданные по умолчанию значения тайм-аутов, то переход на новую конфигурацию может занять свыше 50 секунд: 20 секунд понадобится на констатацию факта потери связи с корневым коммутатором (истечение таймера — единственный способ узнать об этом событии в стандартном варианте STA), еще 2 × 15 секунд нужно для перехода портов в состояние продвижения.

Имеющиеся многочисленные нестандартные версии STA позволяют сократить время реконфигурирования за счет усложнения алгоритма, например добавления новых типов служебных сообщений. В 2001 году была разработана стандартная ускоренная версия протокола — RSTP (спецификация IEEE 802.1w), которая затем вошла в качестве раздела 17 в общий стандарт 802.1D-2004. Для сокращения времени построения активной топологии в версии RSTP использовано несколько новых механизмов и приемов. Назовем их.

Коммутаторы стали учитывать тип сегмента, подключенного к порту. Различаются следующие типы сегментов:

- *Двухточечный сегмент.* В коммутируемых сетях это единственный тип сегмента; для него у порта существует единственный порт-сосед.
- *Разделяемая среда.* Стандарт RSTP по-прежнему учитывает существование разделяемой среды, так как формально ее никто не отменял для скоростей ниже 10 Гбит/с.

- *Тупиковая связь* (edge port). Связь, соединяющая порт коммутатора с конечным узлом сети; по этому сегменту нет смысла ожидать прихода сообщений протокола RSTP. Тупиковая связь конфигурируется администратором.

В случае подключения к порту тупикового сегмента этот порт не участвует в работе протокола RSTP, а сразу после включения переходит в стадию продвижения кадров. Нужно заметить, что в стандарте RSTP начальное заблокированное состояние портов переименовано в состояние отбрасывания. Для портов со связями остальных типов переход в состояние продвижения по-прежнему достижим только после прохождения стадии обучения.

Исключается стадия прослушивания. Коммутаторы не делают паузу в 15 секунд, для того чтобы зафиксировать соответствующую роль порта, например корневого или назначенного. Вместо этого порты переходят в стадию обучения сразу же после назначения им роли корневого или назначенного порта.

Сокращается период фиксации отказа в сети — вместо 10 периодов неполучения сообщений HELLO он стал равен трем таким периодам, то есть 6 секунд вместо 20.

Введены новые роли портов — появились **альтернативный** (alternative) и **резервный** (backup) порты. Альтернативный порт — порт-дублер корневого порта коммутатора, то есть он начинает продвигать кадры в том случае, когда отказывает (либо перестает принимать сообщения HELLO в течение трех периодов) корневой порт. Резервный порт является портом-дублером назначенного порта сегмента; однако такая роль порта имеет смысл только для сегментов, представляющих собой разделяемую среду. Альтернативные и резервные порты находятся в состоянии отбрасывания кадров, так как они не должны продвигать кадры до тех пор, пока их роль не изменится на роль корневого или назначенного порта. Как альтернативные, так и резервные порты выбираются одновременно с корневыми и назначенными портами. Такой подход значительно ускоряет реакцию сети на отказы, так как переход, например, на альтернативный порт происходит сразу же после фиксации отказа и не связан с ожиданием истечения тайм-аутов. За счет новых механизмов и новых ролей портов протокол RSTP строит новую активную топологию быстрее, чем протокол STP, — за несколько секунд вместо одной или нескольких минут. Протокол RSTP совместим с протоколом STP, так что сеть, построенная из коммутаторов, часть из которых поддерживает RSTP, а часть — STP, будет работать нормально.

Фильтрация трафика

Локальная сеть обеспечивает взаимодействие каждого узла с каждым — это очень полезное свойство, так как не требуется производить никаких специальных действий, чтобы обеспечить доступ узла *A* к узлу *B*, — достаточно того, что узлы подключены к одной и той же локальной сети. В то же время в сети могут возникать ситуации, когда такая тотальная доступность узлов нежелательна. Пример — сервер финансового отдела, доступ к которому желательно разрешить только с компьютеров нескольких конкретных сотрудников этого отдела. Доступ можно ограничить и на уровне ОС или системы управления базой данных самого сервера, но для надежности желательно иметь несколько эшелонов защиты и ограничить доступ еще и на уровне сетевого трафика. Многие модели коммутаторов позволяют администраторам задавать, наряду со стандартными условиями их фильтрации в соответствии с информацией адресной таблицы, и дополнительные условия. Такие фильтры называют пользовательскими.

Пользовательский фильтр, который также часто называют **списком доступа** (access list), предназначен для создания дополнительных барьеров на пути кадров, что позволяет ограничивать доступ определенных групп пользователей к отдельным службам сети. Пользовательский фильтр — это набор условий, которые ограничивают обычную логику передачи кадров коммутаторами.

Наиболее простыми являются пользовательские фильтры на основе MAC-адресов станций. MAC-адреса — это та информация, с которой работает коммутатор, позволяя создавать подобные фильтры удобным для администратора способом, возможно, проставляя некоторые условия в дополнительном поле адресной таблицы — например, условие отбрасывать кадры с определенным адресом (см. рис. 10.12 в главе 10). Таким способом пользователю, работающему на компьютере с данным MAC-адресом, полностью запрещается доступ к ресурсам другого сегмента сети. Рассмотрим применение пользовательского фильтра на примере сети, показанной на рис. 11.4. Пусть требуется разрешить доступ к серверу *S*1 только с компьютеров *C*1 и *C*3, причем кадры от всех остальных компьютеров до сервера доходить не должны.

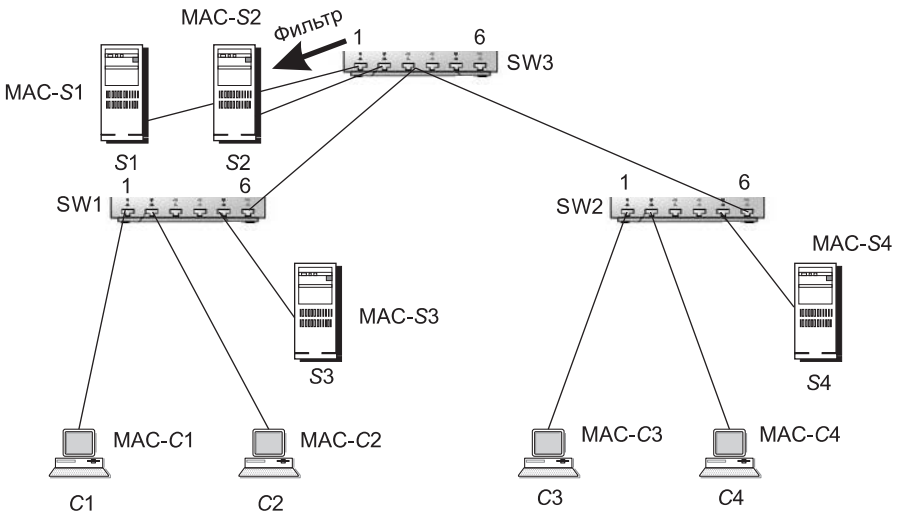


Рис. 11.4. Контроль доступа к серверу с помощью пользовательского фильтра

Список доступа, который решает эту задачу, может выглядеть так:

```
10 permit MAC-C1 MAC-S1
20 permit MAC-C3 MAC-S1
30 deny any any
```

Числа 10, 20 и 30 — это номера строк данного списка. Строки нумеруются с интервалом 10, что дает возможность добавления в этот список других записей, сохраняя исходную последовательность строк. Первое условие разрешает (`permit`) передачу кадра, если его адрес источника равен MAC-C1, а адрес назначения — MA-C-S1; второе условие делает то же, но для кадра с адресом источника MAC-C3; третье условие запрещает (`deny`) передачу кадров с любыми (`any`) адресами.

Чтобы список доступа начал работать, его нужно применить к трафику определенного направления на каком-либо порту коммутатора: или к входящему, или к исходящему. В нашем примере требуется применить список доступа к исходящему трафику порта 1 коммутатора SW3, к которому подключен сервер *S*1. Коммутатор SW3 перед тем, как передать кадр на порт 1, будет просматривать условия списка доступа по очереди. Если какое-то условие из списка соблюдается, то коммутатор выполняет действие этого условия для обрабатываемого кадра и на этом применение списка доступа к данному кадру заканчивается. Поэтому, когда от компьютера *C*1 приходит кадр, адресованный серверу *S*1, то соблюдается первое условие списка, разрешающее передачу кадра, так что коммутатор выполняет стандартное действие по продвижению кадра и тот доходит до сервера *S*1. С кадром от компьютера *C*3 совпадение происходит при проверке второго условия, и он также передается. Однако когда приходят кадры от других компьютеров, например компьютера *C*2, то ни первое, ни второе условия не соблюдаются, зато соблюдается третье условие, поэтому кадр не передается, а отбрасывается.

Списки доступа коммутаторов не работают с широковещательными адресами Ethernet, такие кадры всегда передаются на все порты коммутатора. Отметим, что списки доступа коммутаторов достаточно примитивны, поскольку способны оперировать только информацией канального уровня (MAC-адресами). Списки доступа маршрутизаторов гораздо более гибкие и мощные, поэтому на практике они применяются намного чаще[1].

Иногда администратору требуется задать более тонкие условия фильтрации, например, запретить некоторому пользователю печатать свои документы на Windows-сервере печати, находящемся в чужом сегменте, а остальные ресурсы этого сегмента сделать доступными. Для реализации подобного фильтра нужно запретить передачу кадров, которые удовлетворяют следующим условиям: во-первых, имеют определенный MAC-адрес; во-вторых, содержат в поле данных пакеты SMB; в-третьих, в соответствующем поле этих пакетов в качестве типа сервиса указана печать. Коммутаторы не анализируют протоколы верхних уровней, такие как SMB, поэтому администратору приходится для задания условий фильтрации вручную определять поле, по значению которого нужно осуществлять фильтрацию. В качестве признака фильтрации администратор указывает пару «смещение — размер» относительно начала поля данных кадра канального уровня, а затем приводит еще шестнадцатеричное значение этого поля.

Сложные условия фильтрации обычно записываются в виде булевых выражений, формируемых с помощью логических операторов AND и OR.

Агрегирование линий связи в локальных сетях

Транки и логические каналы

Агрегирование линий связи (физических каналов) между двумя коммуникационными устройствами в один логический канал является еще одной формой использования избыточных альтернативных связей в локальных сетях.

---

[1] Существуют так называемые коммутаторы 3-го уровня, объединяющие функции коммутаторов и маршрутизаторов (в несколько усеченном виде), позволяя фильтровать трафик с привлечением условий на уровне как MAC-адресов, так и IP-адресов.

Отличие техники агрегирования линий связи от алгоритма покрывающего дерева достаточно принципиально.

- Алгоритм STA переводит избыточные связи в горячий резерв, оставляя в рабочем состоянии только минимальный набор линий, необходимых для обеспечения связности сегментов сети. В этом случае *повышается надежность* сети, но не ее производительность.
- При агрегировании физических каналов все избыточные связи остаются в рабочем состоянии, В результате повышается как *надежность* сети, так и ее *производительность*.

При отказе одной из составляющих агрегированного логического канала, который часто называют **транком**, трафик распределяется между оставшимися линиями (рис. 11.5). На рисунке примером такой ситуации является транк 2, в котором один из физических каналов (центральный) отказал, и все кадры передаются по оставшимся двум каналам. Этот пример демонстрирует повышение ***надежности*** при агрегировании.

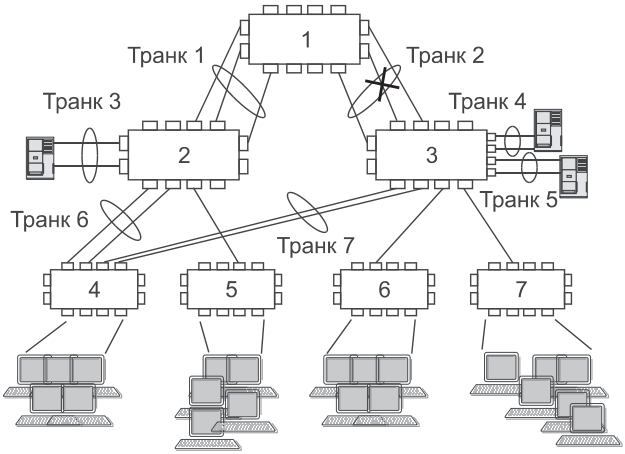


Рис. 11.5. Агрегирование физических каналов

Покажем теперь, как агрегирование линий связи повышает ***производительность*** сети. Так, на рисунке коммутаторы 1 и 3 соединены тремя параллельными линиями связи, что в три раза повышает производительность этого участка сети по сравнению со стандартным вариантом топологии дерева, которая не допускает таких параллельных связей. Повышение производительности связи между коммутаторами путем агрегирования линий связи в некоторых случаях является более эффективным, чем замена единственной линии связи более скоростной. Например, несмотря на то что семейство Ethernet предлагает широкий выбор скоростей физического канала, от 10 Мбит/с до 100–400 Гбит/с, десятикратное повышение скорости при переходе от одного стандарта Ethernet к другому не всегда нужно и экономически оправданно. Например, если в установленных в сети коммутаторах отсутствует возможность добавления модуля с портом 10G Ethernet, то повышение скорости на некоторых каналах до 10 Гбит/с потребует полной замены коммутаторов. В то же время вполне возможно, что у таких коммутаторов имеются свободные порты Gigabit Ethernet, поэтому скорость передачи

данных можно было бы повысить, например, до 6 Гбит/с, объединив в агрегированный канал шесть портов Fast Ethernet.

Агрегирование линий связи используется как для связей между портами коммутаторов локальной сети, так и для связей между компьютером и коммутатором. Чаще всего этот вариант выбирают для высокоскоростных и ответственных серверов. В этом случае все сетевые адаптеры, входящие в транк, принадлежат одному компьютеру и разделяют один и тот же сетевой адрес. Поэтому для протокола IP или другого протокола сетевого уровня порты транка неразличимы, что соответствует концепции единого логического канала, лежащей в основе агрегирования.

Почти все методы агрегирования, применяемые в настоящее время, обладают существенным ограничением — в них учитываются только связи между двумя соседними коммутаторами сети и полностью игнорируется все, что происходит вне этого участка сети. Например, работа транка 1 никак не координируется с работой транка 2, и наличие обычной связи между коммутаторами 2 и 3, которая создает вместе с транками 1 и 2 петлю, не учитывается. Поэтому технику агрегирования линий связи необходимо применять *одновременно* с алгоритмом покрывающего дерева — если администратор сети хочет использовать все топологические возможности объединения узлов сети. Для STA транк должен выглядеть как одна линия связи, тогда логика работы алгоритма останется в силе.

Существует большое количество фирменных реализаций механизма агрегирования линий связи. Наиболее популярные принадлежат, естественно, лидерам в секторе оборудования для локальных сетей. Это такие реализации, как Fast EtherChannel и Gigabit EtherChannel компании Cisco, MultiLink Trunking компании Nortel, Adaptive Load Balancing компании Intel и ряд других. Стандарт IEEE Link Aggregation обобщает эти подходы. Начальные версии этого стандарта были разработаны только для технологии Ethernet, поэтому он разрабатывался рабочей группой 802.3 и получил обозначение 802.3ad, но затем его потенциальная полезность для других технологий была осознана и последние версии этого стандарта находятся в ведении рабочей группы 802.1, последняя версия получила обозначение 802.1AX-2014.

Динамическое агрегирование линий связи в стандарте IEEE Link Aggregation

Стандарт IEEE 802.1AX описывает процедуры агрегирования линий связи между *двумя узлами сети* (называемыми в стандарте системами), связанными между собой линиями связи топологии «точка-точка». Связи другой топологии, например «точка-многоточка», не разрешаются. Возможен также распределенный вариант агрегирования, когда агрегированные линии связи соединяют *два портала* — порталом в стандарте называется группа узлов (состоящая максимум из трех узлов), действующая как единое целое в отношении использования агрегированной линии связи. Далее мы рассмотрим только нераспределенный вариант стандарта.

Стандарт вводит новый **подуровень агрегирования линий связи (Link Aggregation Sublayer, LAS)**, располагающийся между уровнем MAC и уровнем протокола верхнего уровня, который пользуется услугами уровня MAC (рис. 11.6). Таким протоколом может

быть протокол IP, протокол STP или любой другой протокол, которому необходимо передать свои данные с помощью протокола Ethernet. Для протокола верхнего уровня услуги, предоставляемые подуровнем агрегирования линий связи, идентичны услугам уровня MAC, при этом вместо **агрегатной группы портов (Link Aggregation Group, LAG)** они имеют дело с одним портом, называемым **логическим портом агрегатора**.

Логический порт агрегатора имеет свой собственный MAC-адрес (MAC-A на рис. 11.6), который заменяет адрес для протоколов верхнего уровня вместо MAC-адреса физических портов, входящих в группу LAG. То есть в том случае, когда протокол верхнего уровня обращается к протоколу Ethernet с запросом о формировании и отправке нового пакета Ethernet, протокол Ethernet указывает в качестве адреса отправителя адрес MAC-A.

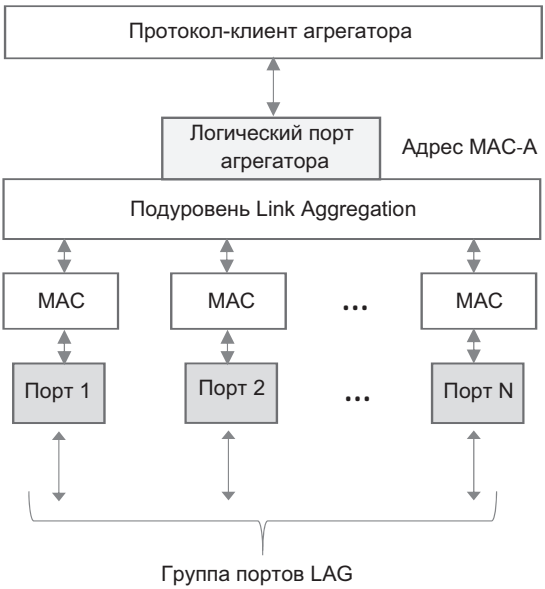


Рис. 11.6. Позиция подуровня агрегирования линий связи в стеке протоколов Ethernet

Естественно, этот MAC-адрес используется только тогда, когда кадры отправляются от имени узла, на котором работает протокол агрегирования линий связи, например, когда агрегируются порты IP-маршрутизатора, который каждый раз упаковывает IP-пакет в новый кадр Ethernet и отправляет своему соседу от своего MAC-адреса (подробно работа IP-маршрутизатора рассматривается в главе 17). Протокол STP также генерирует кадры Ethernet от имени своего узла, поэтому в его кадрах также используется адрес MAC-A. В тех же случаях, когда узел передает кадры Ethernet транзитом, как это делает коммутатор при обработке трафика своих клиентов, MAC-адреса в этих кадрах не изменяются и адрес MAC-A не используется, как не используются MAC-адреса физических портов коммутатора в том случае, когда агрегирование портов не используется.

Алгоритм прозрачного моста работает над подуровнем LAS, поэтому в таблицах продвижения коммутатора фигурирует адрес MAC-A, а не адреса физических портов, входящих в группу портов LAG. Подуровень LAS имеет довольно сложную структуру (рис. 11.7). Основным функциональным блоком подуровня LAS является **агрегатор** — блок, в обязанности которого входит распределение кадров, получаемых от протокола верхнего уровня между физическими портами агрегатной группы LAG, и выполнение обратной операции — агрегирования кадров, поступающих от физических портов в общий поток кадров, передаваемых верхнему уровню.

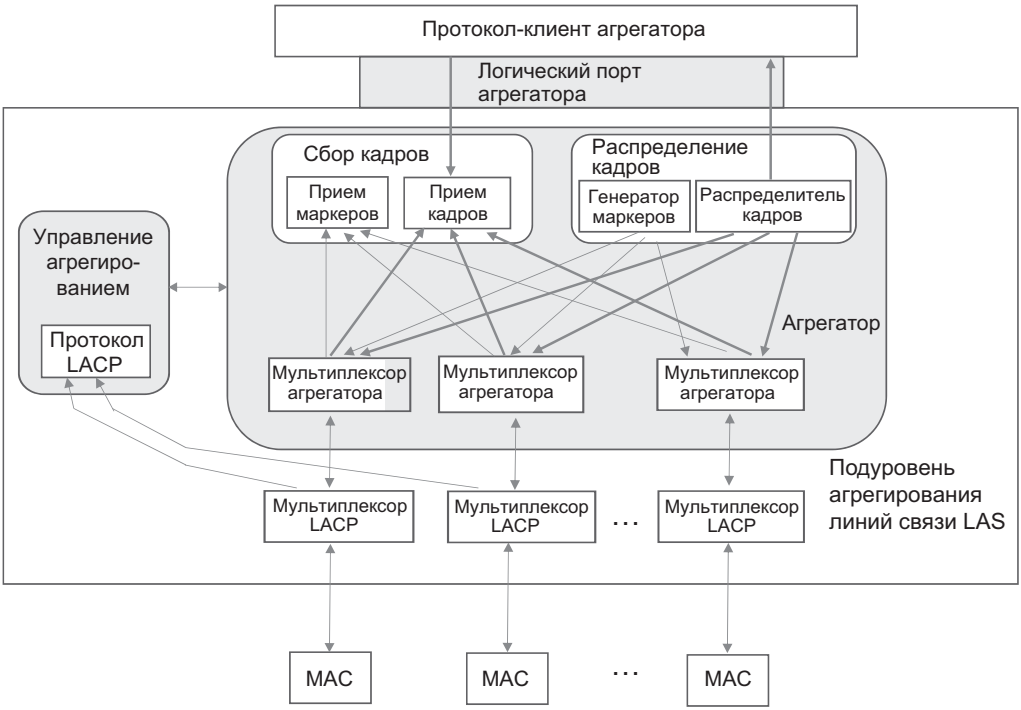


Рис. 11.7. Функциональная структура подуровня агрегирования линий связи

Агрегатор работает под управлением блока управления агрегированием, отвечающим за то, какие физические порты образуют группу LAG. Узел может иметь несколько групп LAG — в этом случае в нем организуется несколько агрегаторов по числу групп. Группа может быть образована как вручную, так и автоматически. В первом случае ее конфигурирует администратор и группа является статической, то есть членство портов в ней не изменяется до тех пор, пока администратор не изменит состав группы. Во втором случае группа образуется автоматически в результате работы **протокола управления агрегированием линий связи LACP** (Link Aggregation Control Protocol).

Протокол LACP выполняет две основные функции:

- образование группы портов LAG в результате переговорного процесса между двумя узлами сети;
- постоянный мониторинг состояния портов и линий связи, входящих в агрегатную группу LAG.

Эти функции взаимосвязаны: если в результате мониторинга выясняется, что порт или линии стали неработоспособными, то состав группы LAG может *динамически* измениться: отказавший порт из нее удаляется, а новый — добавляется. Для выполнения этих функций узлы постоянно обмениваются сообщениями протокола LACP. Описание протокола LACP можно найти на сайте книги.

(S) *Агрегирование линий связи*

Виртуальные локальные сети

Важным свойством коммутатора локальной сети является способность контролировать передачу кадров между сегментами сети. По различным причинам (соблюдение прав доступа, политика безопасности и т. д.) некоторые кадры не следует передавать по адресу назначения.

Ограничения такого типа можно реализовать с помощью *пользовательских фильтров* (см. ранее). Но пользовательский фильтр может запретить коммутатору передачу кадров только по конкретным адресам, а широковещательный трафик он в соответствии с алгоритмом работы *обязан* передать всем сегментам сети. Поэтому сети, созданные на основе коммутаторов, иногда и называют *плоскими* — из-за отсутствия барьеров на пути широковещательного трафика. Технология виртуальных локальных сетей позволяет преодолеть указанное ограничение.

Виртуальной локальной сетью (Virtual Local Area Network, VLAN) называется группа узлов сети, трафик которой, в том числе широковещательный, на канальном уровне полностью изолирован от трафика других узлов сети.

Это означает, что передача кадров между разными виртуальными сетями на основании адреса канального уровня невозможна независимо от типа адреса (уникального, группового или широковещательного). В то же время внутри виртуальной сети кадры передаются по технологии коммутации, то есть только на тот порт, который связан с адресом назначения кадра.

Виртуальные локальные сети могут *перекрываться*, если один или несколько компьютеров входят в состав более чем одной виртуальной сети. На рис. 11.8 сервер электронной почты входит в состав виртуальных сетей 3 и 4. Это означает, что его кадры передаются коммутаторами всем компьютерам, входящим в эти сети. Если же какой-то компьютер входит в состав только виртуальной сети 3, то его кадры до сети 4 доходить не будут, но он сможет взаимодействовать с компьютерами сети 4 через общий почтовый сервер. Такая схема защищает виртуальные сети друг от друга не полностью — например, широковещательный шторм, возникший на сервере электронной почты, затопит и сеть 3, и сеть 4.

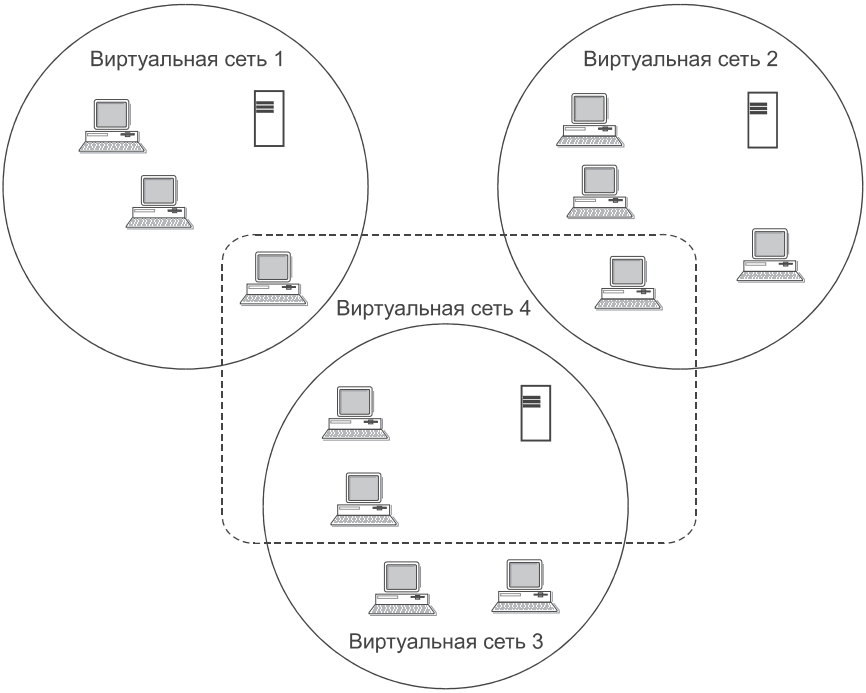


Рис. 11.8. Виртуальные локальные сети

В подобных случаях принято считать, что виртуальная сеть образует *домен широковещательного трафика* по аналогии с доменом коллизий, образуемым повторителями сетей Ethernet.

Назначение виртуальных сетей

Пример из предыдущего раздела демонстрирует, кроме прочего, что с помощью пользовательских фильтров можно вмешиваться в нормальную работу коммутаторов и ограничивать взаимодействие узлов локальной сети в соответствии с требуемыми правилами доступа. Однако механизм пользовательских фильтров коммутаторов имеет несколько недостатков:

- *Приходится задавать отдельные условия для каждого узла сети*, используя при этом громоздкие MAC-адреса. Гораздо проще было бы группировать узлы и описывать условия взаимодействия сразу для групп.
- *Невозможно блокировать широковещательный трафик*. Широковещательный трафик может быть причиной недоступности сети, если какой-то ее узел умышленно или неумышленно с большой интенсивностью генерирует широковещательные кадры.

Техника виртуальных локальных сетей решает задачу ограничения взаимодействия узлов сети другим способом.

Основное назначение технологии VLAN состоит в облегчении процесса создания изолированных сетей, которые затем обычно связываются между собой с помощью маршрутизаторов. Такое построение сети создает мощные барьеры на пути нежелательного трафика из одной сети в другую. Сегодня считается очевидным, что любая крупная сеть должна включать маршрутизаторы, иначе потоки ошибочных кадров, например широковещательных, будут периодически «затапливать» всю сеть через прозрачные для них коммутаторы, приводя ее в неработоспособное состояние.

Достоинством технологии виртуальных сетей является то, что она позволяет создавать полностью изолированные сегменты сети путем логического конфигурирования коммутаторов, не прибегая к изменению физической структуры.

До появления технологии VLAN для создания отдельной сети использовались либо физически изолированные сегменты коаксиального кабеля, либо не связанные между собой сегменты, построенные на повторителях и мостах. Затем эти сети связывались маршрутизаторами в единую составную сеть (рис. 11.9).

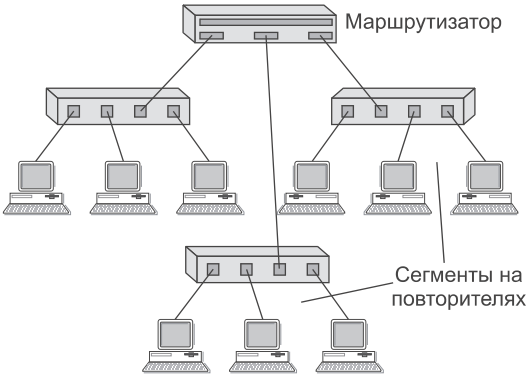


Рис. 11.9. Составная сеть, состоящая из сетей, построенных на основе повторителей

Изменение состава сегментов (переход пользователя в другую сеть, дробление крупных сегментов) при таком подходе подразумевает физическую перекоммутацию разъемов на передних панелях повторителей или на кроссовых панелях, что не очень удобно в больших сетях — это требует объемной физической работы, к тому же высока вероятность ошибки.

Для связывания виртуальных сетей в общую сеть требуется привлечение средств сетевого уровня. Он может быть реализован в отдельном маршрутизаторе или в составе программного обеспечения коммутатора, который тогда становится комбинированным устройством — так называемым **коммутатором 3-го уровня**.

Технология виртуальных сетей долгое время не стандартизировалась, хотя и была реализована в очень широком спектре моделей коммутаторов разных производителей. Положение изменилось после принятия в 1998 году стандарта IEEE 802.1Q, который определяет базовые правила построения виртуальных локальных сетей, не зависящие от протокола канального уровня, поддерживаемого коммутатором.

Создание виртуальных сетей на базе одного коммутатора

При создании виртуальных сетей на основе одного коммутатора обычно используется механизм *группирования портов* коммутатора (рис. 11.10). При этом каждый порт приписывается той или иной виртуальной сети. Кадр, пришедший от порта, принадлежащего, например, виртуальной сети 1, никогда не будет передан порту, который не принадлежит этой виртуальной сети. Впрочем, порт можно приписать нескольким виртуальным сетям, хотя на практике так делают редко — пропадает эффект полной изоляции сетей.

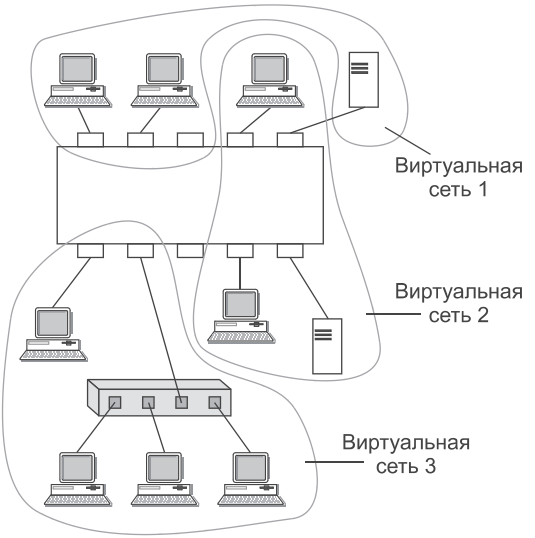


Рис. 11.10. Виртуальные сети, построенные на одном коммутаторе

Создание виртуальных сетей путем группирования портов не требует от администратора большого объема ручной работы — достаточно каждый порт приписать к одной из нескольких заранее поименованных виртуальных сетей. Обычно такая операция выполняется с помощью специальной программы, прилагаемой к коммутатору.

Второй способ образования виртуальных сетей основан на *группировании MAC-адресов*. Каждый MAC-адрес, который изучен коммутатором, приписывается той или иной виртуальной сети. При существовании в сети множества узлов этот способ требует от администратора большого объема ручной работы и по этой причине не получил распространения.

Создание виртуальных сетей на базе нескольких коммутаторов

Рисунок 11.11 иллюстрирует проблему, возникающую при создании виртуальных сетей на основе нескольких коммутаторов, поддерживающих технику *группирования портов*.

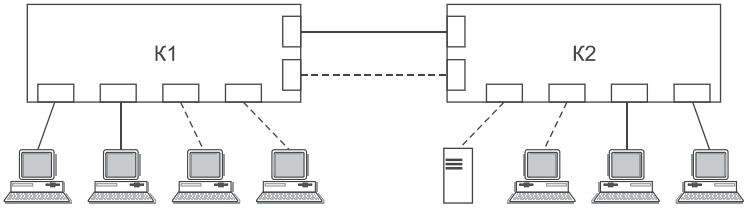


Рис. 11.11. Построение виртуальных сетей на нескольких коммутаторах с группированием портов

Если узлы какой-либо виртуальной сети подключены к разным коммутаторам, то для подключения каждой такой сети на коммутаторах должна быть выделена специальная пара портов. В противном случае, если коммутаторы будут связаны только одной парой портов, информация о принадлежности кадра той или иной виртуальной сети при передаче из коммутатора в коммутатор будет утеряна. Таким образом, коммутаторы с группированием портов требуют для своего соединения столько портов, сколько виртуальных сетей они поддерживают. Порты и кабели используются в этом случае очень расточительно. Кроме того, при соединении виртуальных сетей через маршрутизатор для каждой виртуальной сети выделяются отдельные кабель и порт маршрутизатора, что также приводит к большим накладным расходам.

Группирование MAC-адресов в виртуальную сеть на каждом коммутаторе избавляет от необходимости связывать их по нескольким портам, поскольку в этом случае MAC-адрес становится меткой виртуальной сети. Однако этот способ требует выполнения большого количества ручных операций по маркировке MAC-адресов на каждом коммутаторе сети.

Оба подхода, следовательно, основаны только на добавлении дополнительной информации к адресным таблицам коммутатора и в них отсутствует возможность встраивания в передаваемый кадр информации о принадлежности кадра виртуальной сети. Поэтому широкое распространение получил иной подход, основанный на введении в кадр *дополнительного поля*, которое хранит информацию о принадлежности кадра той или иной виртуальной локальной сети при его перемещениях между коммутаторами сети. При этом нет необходимости помнить в каждом коммутаторе о принадлежности всех MAC-адресов составной сети виртуальным сетям. Дополнительное поле с пометкой о номере виртуальной сети используется только тогда, когда кадр передается от коммутатора к коммутатору, а при передаче кадра конечному узлу оно обычно удаляется. При этом модифицируется только протокол взаимодействия «коммутатор-коммутатор», а программное и аппаратное обеспечение конечных узлов остается неизменным.

До принятия стандарта IEEE 802.1Q существовало много фирменных протоколов этого типа, но все они имели один недостаток — оборудование различных производителей при образовании VLAN оказывалось несовместимым. Стандарт IEEE 802.1Q вводит в кадре Ethernet дополнительный заголовок — тег виртуальной локальной сети.

Тег виртуальной локальной сети состоит из поля **TCI** (Tag Control Information — управляющая информация тега) размером в 2 байта и предшествующего ему поля **EtherType**, которое является стандартным для кадров Ethernet и также состоит из 2 байтов (рис. 11.12).

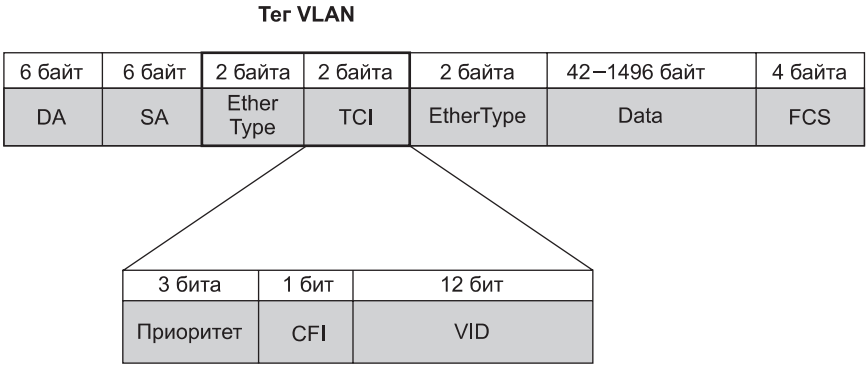


Рис. 11.12. Структура помеченного кадра Ethernet

Тег VLAN не является обязательным для кадров Ethernet. Кадр, у которого имеется такой заголовок, называют **помеченным** (tagged frame). Коммутаторы могут одновременно работать как с помеченными, так и с непомеченными кадрами. Из-за добавления тега VLAN максимальная длина поля данных уменьшилась на 4 байта.

Чтобы оборудование локальных сетей могло отличать и понимать помеченные кадры, для них введено специальное значение поля EtherType, равное 0x8100. Это значение говорит о том, что за ним следует поле TCI, а не стандартное поле данных. Обратите внимание, что в помеченном кадре за полями тега VLAN следует другое поле EtherType, указывающее тип протокола, данные которого переносятся полем данных кадра.

В поле TCI находится 12-битное поле номера (идентификатора) VLAN, называемого *VID*. Разрядность поля VID позволяет коммутаторам создавать до 4096 виртуальных сетей. Помимо этого, в поле TCI помещено трехбитное поле *приоритета* кадра. Однобитное поле *CFI* было введено с целью поддержания специального формата кадра Token Ring, для сетей Ethernet оно должно содержать значение 0. Пользуясь значением VID в помеченных кадрах, коммутаторы сети выполняют групповую фильтрацию трафика, разбивая сеть на виртуальные сегменты, то есть на VLAN. Для поддержки этого режима каждый порт коммутатора приписывается к одной или нескольким виртуальным локальным сетям, то есть выполняется группировка портов. Поле приоритета предназначено для согласованного обеспечения качества обслуживания (QoS) различных классов трафика. Всего может поддерживаться до 8 классов трафика (это определяется тремя битами поля), при этом коммутаторы могут применять все методы обеспечения QoS, описанные в главе 5.

Конфигурирование VLAN

Существуют различные подходы к конфигурированию виртуальных локальных сетей, построенных на нескольких коммутаторах.

Схема «транк — линия доступа»

Наиболее распространенным является подход, основанный на понятиях линии доступа и транка.

Линия доступа связывает порт коммутатора (называемый в этом случае **портом доступа**) с конечным узлом (компьютером, мобильным устройством и т. п.), принадлежащим некоторой виртуальной локальной сети. Предполагается, что конечный узел работает с непомеченными кадрами, то есть структура VLAN для него прозрачна.

Транк — это линия связи, которая соединяет между собой порты двух коммутаторов; в общем случае через транк передается трафик нескольких виртуальных сетей.

Коммутаторы, поддерживающие технику VLAN, без специального конфигурирования по умолчанию работают как стандартные коммутаторы, обеспечивая соединения всех со всеми. В сети, образованной такими коммутаторами, все конечные узлы по умолчанию относятся к условной сети VLAN1 с идентификатором VID, равным 1. Все порты этой сети, к которым подключены конечные узлы, по определению являются портами доступа. Сеть VLAN1 можно отнести к виртуальным локальным сетям лишь *условно*, так как по ней передаются непомеченные кадры. Условная сеть VLAN также называется сетью VLAN, предлагаемой по умолчанию (default VLAN), или естественной (native VLAN).

Чтобы образовать в исходной сети виртуальную локальную сеть, нужно в первую очередь выбрать для нее значение идентификатора VID, отличное от 1, а затем, используя команды конфигурирования коммутатора, приписать к этой сети те порты, к которым присоединены включаемые в нее компьютеры. Порт доступа может быть приписан только к одной виртуальной локальной сети.

Порты доступа получают от конечных узлов сети непомеченные кадры, маркируя их тегом VLAN, содержащим то значение VID, которое назначено этому порту. При передаче же помеченных кадров конечному узлу порт доступа удаляет тег виртуальной локальной сети. Для более наглядного описания вернемся к рассмотренному ранее примеру сети. На рис. 11.13 показано, как решается задача избирательного доступа к серверам на основе техники VLAN.

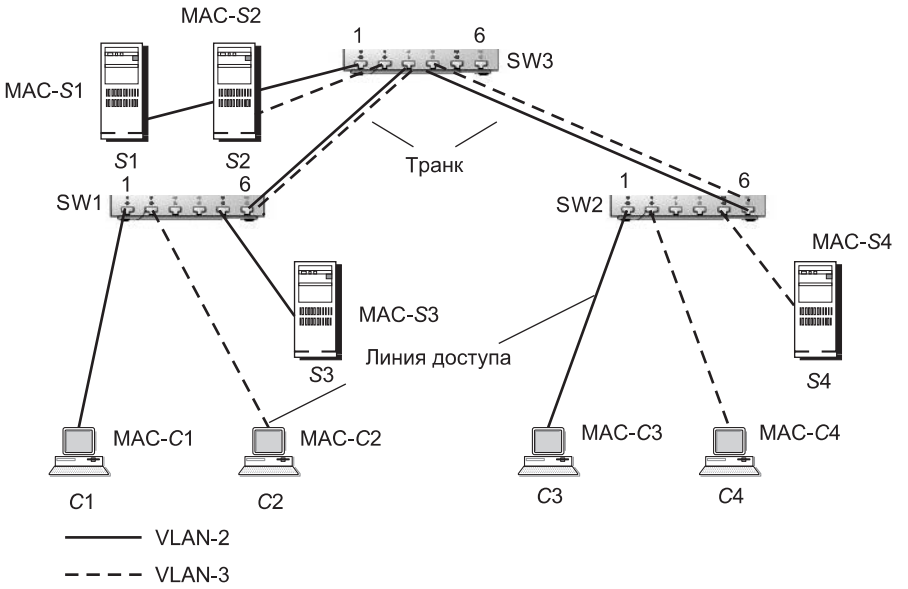


Рис. 11.13. Разбиение сети на две виртуальные локальные сети

Будем считать, что поставлена задача обеспечить доступ компьютеров *C*1 и *C*3 к серверам *S*1 и *S*3, в то время как компьютеры *C*2 и *C*4 должны иметь доступ только к серверам *S*2 и *S*4. Чтобы решить эту задачу, можно организовать две виртуальные локальные сети, VLAN2 и VLAN3 (напомним, что сеть VLAN1 уже существует по умолчанию — это наша исходная сеть), приписав один набор компьютеров и серверов к VLAN2, а другой — к VLAN3. Первым шагом в конфигурировании VLAN2 и VLAN3 является их активизация в каждом из коммутаторов сети.

Затем к этим сетям VLAN можно приписать определенные порты, работающие в режиме доступа. Для приписывания конечных узлов к определенной виртуальной локальной сети соответствующие порты объявляются портами доступа этой сети путем назначения им соответствующего идентификатора VID. Например, порт 1 коммутатора SW1 должен быть объявлен портом доступа VLAN2 путем назначения ему идентификатора VID2, то же самое должно быть проделано с портом 5 коммутатора SW1, портом 1 коммутатора SW2 и портом 1 коммутатора SW3. Порты доступа сети VLAN3 должны получить идентификатор VID3.

В нашей сети нужно также организовать транки — те линии связи, которые соединяют между собой порты коммутаторов. Порты, подключенные к транкам, не добавляют и не удаляют теги, а просто передают кадры в неизменном виде. В нашем примере такими портами должны быть порты 6 коммутаторов SW1 и SW2, а также порты 3 и 4 коммутатора SW3. Порты в нашем примере должны поддерживать сети VLAN2 и VLAN3 (и VLAN1, если в сети есть узлы, явно не приписанные ни к одной виртуальной локальной сети). Транк может быть сконфигурирован как в «неразборчивом» режиме, когда он передает кадры с любым номером VLAN, так и в избирательном режиме, когда он передает кадры только определенных номеров VLAN.

Коммутаторы, поддерживающие технологию VLAN, осуществляют дополнительную фильтрацию трафика. В том случае, если таблица продвижения коммутатора говорит о том, что пришедший кадр нужно передать на некоторый порт, перед передачей коммутатор проверяет, соответствует ли значение VID в теге VLAN кадра той виртуальной локальной сети, которая приписана к этому порту. В случае соответствия кадр передается, несоответствия — отбрасывается. Непомеченные кадры обрабатываются аналогичным образом, но с использованием условной сети VLAN1. MAC-адреса изучаются коммутаторами сети отдельно по каждой виртуальной локальной сети.

Схема с гибким конфигурированием портов

В этой схеме порты не делятся на транки и порты доступа, каждый порт может быть гибко сконфигурирован для специфической поддержки кадров VLAN в зависимости от потребностей сети. Порт может работать в следующих режимах:

- *Принимать только непомеченные кадры.* В этом случае режим соответствует режиму порта доступа.
- *Принимать только помеченные кадры.* При этом порту могут быть приписаны один или несколько номеров VLAN. Этот режим соответствует избирательному режиму работы транка. Помеченные кадры передаются без отбрасывания/добавления тега VLAN.
- *Принимать как помеченные, так и непомеченные кадры.* Непомеченные кадры всегда принадлежат естественной сети VLAN1 (некоторые модели коммутаторов позволяют

администратору назначить естественной сети VLAN произвольный номер, отличный от 1). Порту может быть приписан один или несколько номеров VLAN.

В том случае, когда коммутатор поддерживает образование нескольких логических портов для одного и того же физического порта, каждый логический порт может работать в собственном режиме.

В схеме с гибким конфигурированием портов администратору проще производить изменения в конфигурации виртуальных локальных сетей, так как ему не требуется изменять роли портов в сети (например, изменять роль порта доступа на роль транка), но при этом увеличивается объем конфигурационных операций.

Автоматизация конфигурирования VLAN

В сети, состоящей из большого количества коммутаторов и не разделенной на подсети маршрутизаторами, полностью ручное конфигурирование VLAN может приводить к ошибкам из-за несогласованности информации об активных сетях VLAN на различных коммутаторах, особенно если их конфигурируют разные администраторы.

Существует несколько протоколов, позволяющих частично автоматизировать конфигурирование VLAN в сети.

Cisco VLAN Trunking Protocol (VTP). Этот протокол является фирменным протоколом компании Cisco и работает только на ее коммутаторах. Коммутаторы Cisco поддерживают модель «транк — линии доступа», а протокол VTP позволяет по транковым связям передавать информацию о сетях VLAN, активизированных на одном из коммутаторов, другим коммутаторам сети. Поэтому администратору достаточно добавить (или удалить) VLAN на одном из коммутаторов сети, после чего все остальные коммутаторы сети получат информацию о добавлении (удалении) VLAN с данным номером и произведут соответствующие изменения в своих конфигурационных записях.

Для удобства администрирования больших сетей в протоколе VTP существует понятие домена — все сообщения протокола VTP воспринимаются коммутаторами только одного и того же домена (имя домена и его пароль конфигурируются на каждом коммутаторе вручную). Приписывание VLAN порту доступа при работе протокола VTP по-прежнему выполняется вручную. Порты, работающие в режиме транка, приписывают номера VLAN к транку (работающему в избирательном режиме) динамически. При этом протокол VTP автоматически выполняет отсечение номера VLAN для транков некоторого домена, если в данном домене этот номер не приписан ни одному из его портов доступа (это свойство называется VTP pruning).

Свои VTP-объявления коммутаторы рассылают с использованием группового адреса.

GARP VLAN Registration Protocol (*GVRP*). Этот протокол является одним из двух популярных приложений протокола GARP (Generic Attribute Registration Protocol). Протокол GARP был разработан рабочей группой IEEE 802.1 для того, чтобы коммутаторы локальной сети могли сообщать друг другу (регистрировать в сети) различные атрибуты. На практике этот протокол стали применять для регистрации двух типов атрибутов: достижимых через некоторый порт коммутатора групповых MAC-адресов и номеров VLAN. Соответственно, появились два приложения протокола GARP: GMRP (GARP Multicast Registration Protocol) и GVRP (GARP VLAN Registration Protocol). Так, приложение

GMRP позволяет коммутаторам отсечь бесполезный трафик с групповыми адресами от сегментов сети, в которых нет активных получателей этих адресов.

Назначение приложения GVRP примерно то же, что у протокола Cisco VTP. Он позволяет конфигурировать новую сеть VLAN только на одном из коммутаторов большой сети — остальные коммутаторы выполняют изменения в своей конфигурации автоматически, получая сообщения GVRP. GVRP является стандартным протоколом и поэтому работает на коммутаторах различных производителей, в отличие от фирменного протокола Cisco VTP. Кроме того, в отличие от VTP, он может работать не только на портах-транках, но и на портах доступа, к тому же конечные узлы также могут поддерживать GVRP, а значит, сетевой адаптер компьютера может инициировать динамическое приписывание номера своей сети VLAN у порта доступа.

Multiple VLAN Registration Protocol (*MVRP*). Протокол GARP обладал несколькими существенными недостатками — в больших сетях он порождал большое количество служебного трафика, кроме того, процесс установления новой конфигурации мог длиться слишком долго из-за нескольких обязательных тайм-аутов. Поэтому в 2007 году группа IEEE 802.1 заменила GARP протоколом MRP (Multiple Registration Protocol). Соответственно, протокол MVRP заменил GVRP (а MMRP — GMRP). За счет изменения формата сообщений и логики обмена ими служебный трафик был сокращен, а время установления новой конфигурации уменьшено.

Альтернативные маршруты в виртуальных локальных сетях

По умолчанию протокол STP/RSTP образует в сети одно покрывающее дерево для всех виртуальных локальных сетей. Чтобы в сети можно было использовать разные покрывающие деревья для разных виртуальных локальных сетей, существует специальная версия протокола, называемая **множественным протоколом покрывающего дерева** (Multiple Spanning Tree Protocol, **MSTP**).

Протокол MSTP позволяет создать несколько покрывающих деревьев и приписывать к ним различные виртуальные локальные сети. Обычно создается небольшое количество деревьев, например два или три, чтобы сбалансировать нагрузку на коммутаторы, в противном случае, как мы видели в примере на рис. 11.2 и 11.3, единственное покрывающее дерево может полностью оставить без работы некоторые коммутаторы сети, то есть недоиспользовать имеющиеся сетевые ресурсы. Если вернуться к ранее рассмотренному примеру (см. рис. 11.2), то при создании двух покрывающих деревьев можно сконфигурировать приоритеты коммутаторов так, чтобы для одного дерева корневым коммутатором стал коммутатор 111, а для второго — коммутатор 222 (рис. 11.14).

В этом варианте мы подразумеваем, что порты четырех коммутаторов с 555 по 888 сконфигурированы как порты доступа одной виртуальной локальной сети, например VLAN100, а порты трех тех же коммутаторов — как порты доступа другой виртуальной локальной сети, например VLAN200. Сеть VLAN100 приписана к покрывающему дереву с корневым коммутатором 111, а VLAN200 — к покрывающему дереву с корневым коммутатором 222. В этом варианте все коммутаторы сети используются для передачи трафика, что повышает производительность сети. Протокол MSTP основан на протоколе RSTP, поэтому обеспечивает быструю реакцию сети на отказы.

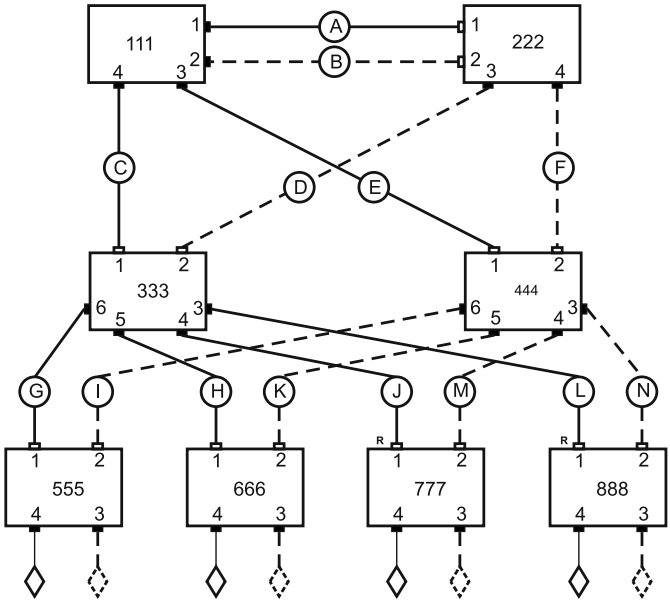


Рис. 11.14. Два покрывающих дерева, построенные по протоколу MSTP

Ограничения коммутаторов

Применение коммутаторов позволяет преодолеть ограничения, свойственные сетям с разделяемой средой. Коммутируемые локальные сети могут покрывать значительные территории, плавно переходя в сети мегаполисов; они могут состоять из сегментов различной пропускной способности, образуя сети с очень высокой производительностью; они могут использовать альтернативные маршруты для повышения надежности и производительности. Но построение сложных сетей без маршрутизаторов, лишь на основе коммутаторов, имеет существенные ограничения.

- Серьезные ограничения по-прежнему накладываются на топологию коммутируемой локальной сети. Требование *отсутствия петель* преодолевается с помощью техники STP/RSTP/MSTP и агрегирования каналов лишь частично. Действительно, STP не позволяет задействовать все альтернативные маршруты для передачи пользовательского трафика, а агрегирование каналов разрешает так делать только на участках сети между двумя соседними коммутаторами. Подобные ограничения не позволяют применять многие эффективные топологии, пригодные для передачи трафика.
- Логические сегменты сети, расположенные между коммутаторами, *слабо изолированы* друг от друга, а именно — не защищены от так называемых широковещательных штормов. Использование же механизма виртуальных сетей, реализованного во многих коммутаторах, хотя и позволяет достаточно гибко создавать изолированные по трафику группы станций, изолирует их полностью, то есть так, что узлы одной виртуальной сети не могут взаимодействовать с узлами другой виртуальной сети.
- В сетях, построенных на основе мостов и коммутаторов, достаточно *сложно решается задача фильтрации трафика* на основе данных, содержащихся в пакете. В таких сетях

фильтрация выполняется только с помощью пользовательских фильтров, для создания которых администратору приходится иметь дело с двоичным представлением содержимого пакетов.

- Реализация транспортной подсистемы только средствами физического и канального уровней приводит *к недостаточно гибкой одноуровневой системе адресации*: в качестве адреса назначения используется MAC-адрес, жестко связанный с сетевым адаптером.
- У коммутаторов *ограничены возможности по трансляции протоколов* при создании гетерогенной сети. Они не могут транслировать протоколы WAN в протоколы LAN из-за различий в системе адресации этих сетей, а также различных значений максимального размера поля данных.

Наличие серьезных ограничений у протоколов канального уровня показывает, что построение на основе средств этого уровня больших неоднородных сетей является весьма проблематичным. Естественное решение в этих случаях — привлечение средств более высокого сетевого уровня.

ГЛАВА 12 Ethernet операторского класса

Движущие силы экспансии Ethernet

Классическая технология Ethernet разрабатывалась исключительно как технология локальных сетей, и последние до недавнего времени были единственной областью ее применения. Но бесспорный успех технологии Ethernet в локальных сетях, где она вытеснила все иные технологии, привел к напрашивающейся идее о ее использовании в глобальных сетях, преимущественно являющихся операторскими. Обозначим потенциальные преимущества от экспансии Ethernet за пределы локальных сетей.

Для пользователей технология Ethernet важна *в качестве услуги глобальных сетей*. Заметим, эта услуга может у разных провайдеров называться по-разному — Carrier Ethernet, Ethernet VPN, VPLS, ELINE или ELAN, — но суть от этого не меняется: пользователи получают возможность соединения своих территориально рассредоточенных сетей, подключая их к интерфейсу Ethernet, предоставляемому провайдером. При этом их сети объединяются так же, как они объединяются в пределах офиса, то есть на уровне Ethernet и без привлечения протокола IP. Это означает, что сеть провайдера учитывает только MAC-адреса, идентификаторы VLAN и физический интерфейс пользователя, для того чтобы надлежащим образом обеспечить объединение сетей пользователя. При этом пользователи имеют дело с хорошо изученной технологией на интерфейсах доступа к сети провайдера, то есть интерфейсах UNI. Кроме того, при соединении сетей на канальном уровне пользователи свободны в IP-адресации своих сетей, так как при передаче трафика между сетями пользователей услуги Ethernet операторского класса провайдер не применяет IP-адреса. Таким образом, можно, например, назначить адреса одной и той же IP-подсети для всех сетей пользователей или применить частные IP-адреса.

Для провайдеров технология Ethernet операторского класса важна не только как популярная услуга, но и как *внутренняя транспортная технология канального уровня*, в этом случае ее также называют **Carrier Ethernet Transport** и могут использовать для реализации глобальных услуг Ethernet или же создания надежных, быстрых и контролируемых соединений между маршрутизаторами со скоростями до 400 Гбит/с.

Привлекательность Ethernet как внутренней транспортной технологии для операторов связи объясняется относительно низкой стоимостью оборудования Ethernet. Порты Ethernet всегда обладали самой низкой стоимостью по сравнению с портами любой другой технологии (естественно, с учетом скорости передачи данных портом). Низкая стоимость изначально была результатом простоты технологии Ethernet, которая предлагает только минимальный набор функций по передаче кадров в режиме доставки по возможности (с максимальными усилиями), не поддерживая ни контроль над маршрутами трафика, ни мониторинг работоспособности соединения между узлами. Низкая стоимость оборудования Ethernet при удовлетворительной функциональности привела к доминированию

технологии на рынке оборудования локальных сетей, а далее начал работать механизм положительной обратной связи: хорошие продажи — массовое производство — еще большее удешевление — и т. д.

Стремление к унификации также относится к силам, ведущим к экспансии Ethernet в глобальные сети. Сетевой уровень уже давно демонстрирует однородность благодаря доминированию протокола IP, и перспектива получить однородный канальный уровень в виде Ethernet выглядит очень заманчивой.

Однако все это относится к области желаний, а как обстоит дело с возможностями? Готова ли технология Ethernet к новой миссии? Ответ очевиден — в своем классическом виде технологии локальной сети не готова. Чтобы успешно работать в сетях операторов связи, технология и воплощающее ее оборудование должны обладать определенным набором характеристик, среди которых в первую очередь выделяются надежность, отказоустойчивость, масштабируемость и управляемость.

Области улучшения Ethernet

Рассмотрим более подробно те новые свойства, которые необходимо добавить к классическому варианту Ethernet, чтобы превратить Ethernet в транспортную технологию операторского класса, способную работать в сети провайдера в качестве основного транспортного механизма.

Разделение адресных пространств пользователей и провайдера

Адресное пространство сети современной коммутируемой сети Ethernet состоит из двух частей: значений MAC-адресов конечных узлов и значений идентификаторов локальных виртуальных сетей (VLAN), на которые логически разделена сеть. Коммутаторы Ethernet при принятии решения о продвижении кадра учитывают оба адресных параметра.

Если сеть провайдера будет составлять с сетями пользователей единое целое на уровне Ethernet, то такая сеть окажется практически неработоспособной, так как все коммутаторы провайдера должны будут в своих таблицах продвижения содержать MAC-адреса всех конечных узлов всех пользователей, а также поддерживать принятое каждым пользователем разбиение сети на локальные виртуальные сети. Помимо очевидной проблемы количества MAC-адресов (для крупного провайдера это значение может доходить до нескольких миллионов), есть еще и проблема их уникальности — хотя система назначения адресов и призвана предотвратить дублирование «аппаратных» MAC-адресов, существуют еще и программируемые адреса, да и ошибки в прошивке аппаратных адресов тоже случаются.

Применение в сети провайдера пользовательских идентификаторов VLAN также приводит к проблемам. Пользователям нужно договариваться о согласованном применении идентификаторов VLAN, чтобы они были уникальными для каждого пользователя, так как только тогда сеть провайдера сможет доставлять кадры нужным пользовательским сетям. Представить, как реализовать такую процедуру практически, очень непросто, ведь каждый новый пользователь приходит со своими значениями идентификаторов VLAN, и если заставлять его их переназначать, то можно потерять пользователя. Кроме того,

стандарт VLAN изначально не был рассчитан на глобальное применение и поэтому в нем предусмотрено только 4092 значения метки, что крайне мало для крупного провайдера.

Если посмотреть, как решаются эти проблемы в сетях провайдеров, построенных на других принципах, то мы увидим, что при применении провайдером технологии IP MAC-адреса пользователей вообще не проникают в маршрутизаторы провайдера[1], а IP-адреса пользователей представлены в таблицах маршрутизаторов в агрегированном виде — прием, недоступный для плоских MAC-адресов.

Маршрутизация, инжиниринг трафика и отказоустойчивость

Операторы связи привыкли к ситуации полного контроля над путями следования трафика в своих сетях, что обеспечивают, например, технологии SDH и OTN. В IP-сетях степень контроля оператора над маршрутами трафика очень низкая, и одной из причин популярности технологии MPLS (см. главу 20) служит то, что она привнесла в IP-сети возможности инжиниринга трафика. Другой желательной для операторов характеристикой сети является отказоустойчивость маршрутов, то есть возможность быстрого перехода на новый маршрут при отказах узлов или линий связи сети. Технологии SDH и OTN всегда были в этом плане эталоном, обеспечивая переход с основного на заранее проложенный резервный путь за десятки миллисекунд.

В сетях Ethernet маршрутизация трафика и отказоустойчивость обеспечиваются протоколом покрывающего дерева (STP). Этот протокол дает администратору сети очень ограниченный контроль над выбором маршрута (это справедливо и для новых вариантов STP, таких как RSTP и MSTP). Кроме того, покрывающее дерево является общим для всех потоков независимо от их адреса назначения. Ввиду этих особенностей протокол STP/RSTP является очень плохим решением в отношении инжиниринга трафика. И хотя STP обеспечивает отказоустойчивость маршрутов, причем новая версия RSTP значительно сократила время переключения на новый маршрут (с нескольких десятков секунд до одной-двух), до миллисекундного диапазона SDH ей очень далеко. Все это требует нового подхода к маршрутизации потоков в сетях CET, и IEEE работает над этой проблемой.

Функции эксплуатации, администрирования и обслуживания

Функции эксплуатации, администрирования и обслуживания (Operation, Administration, Maintenance, **OAM**) всегда были слабым звеном Ethernet, и это одна из главных причин, по которой операторы связи не хотели применять эту технологию в своих сетях. Новые стандарты, предлагаемые IEEE и ITU-T, призваны исправить эту ситуацию, вводя средства, с помощью которых можно выполнять мониторинг достижимости узлов, локализовывать неисправные сегменты сети и измерять уровень задержек и потерь кадров между узлами сети.

---

[1] Если быть педантичным, то нужно сделать оговорку: за исключением MAC-адресов пограничных интерфейсов пользовательских маршрутизаторов, которые попадают в ARP-таблицы интерфейсов пограничных маршрутизаторов провайдера в случае, если это интерфейсы Ethernet.

Подытоживая, отметим, что первая область связана с решением проблемы использования Ethernet для оказания пользовательских услуг, а две остальные — с приданием Ethernet функциональности, необходимой для применения Ethernet в качестве внутренней транспортной технологии оператора связи.

Функции OAM в Ethernet операторского класса

Начнем рассмотрение улучшений Ethernet с группы функций OAM. К настоящему времени разработано несколько стандартов, относящихся к функциям эксплуатации, администрирования и обслуживания, необходимых для превращения Ethernet в Ethernet операторского класса:

- IEEE 802.1ag. Connectivity Fault Management (CFM). Стандарт описывает протокол мониторинга состояния соединений.
- ITU-T Y.1731. Стандарт комитета ITU-T воспроизводит функции стандарта IEEE 802.1ag CFM, расширяя их за счет группы функций мониторинга параметров QoS.
- IEEE 802.3ah. Стандарт тестирования физического соединения Ethernet.
- MEF E-LMI. Интерфейс локального управления Ethernet.

Протокол CFM

Протокол CFM обеспечивает мониторинг логических соединений Ethernet и ориентируется на технику виртуальных локальных сетей (VLAN). Под логическим соединением в нем понимается соединение узлов, принадлежащих одной сети VLAN. Протокол CFM рассчитан на тестирование соединений любой топологии (двухточечной, звездообразной, полносвязной) и может выполнять мониторинг как в сети, принадлежащей одному провайдеру (однодоменный сценарий), так и в тех случаях, когда соединение проходит через сети нескольких провайдеров (многодоменный сценарий). Мониторинг выполняется между так называемыми **конечными точками обслуживания** (Maintenance End Point, **MEP**), представляющими собой конечные точки соединения, состояние которого нужно наблюдать. Точки MEP располагаются на интерфейсах коммутаторов сети Ethernet, то есть мониторинг выполняется между двумя интерфейсами коммутаторов сети.

Каждая из точек MEP периодически посылает **сообщения проверки непрерывности соединения** (Continuity Check Message, **CCM**), оформленные как кадры сети VLAN, соединения которой тестируются. Например, если наблюдается соединение VLAN 5, то сообщения CCM оформляются как кадры Ethernet с идентификатором VLAN, равным 5. Соединение между точками MEP тестируется отдельно в каждом направлении. Мониторинг CFM осуществляется путем *активных измерений*, так как для его реализации генерируются служебные сообщения CCM, а не используются кадры пользовательского трафика. Устройства, которые не имеют точек MEP, передают сообщения CCM транзитом. Если некоторая точка MEP не принимает сообщений CCM от другой точки MEP в течение заданного *тайм-аута*, то соединение считается *неработоспособным*.

В промежуточных устройствах, через которые проходит соединение, можно сконфигурировать **промежуточные точки обслуживания** (Maintenance Intermediate Point, **MIP**).

Эти точки помогают локализовать проблему, собирая статистику о проходящих через них транзитом сообщениях CCM, сами они такие сообщения не генерируют. Помощь MIP состоит в том, что при наличии проблемы (то есть в том случае, когда сообщения CCM не проходят от одной точки MEP до другой) факт прохождения сообщений CCM через некоторую точку MIP говорит о том, что данный сегмент сети работоспособен и причину проблемы нужно искать в другом сегменте. На рис. 12.1 показан пример мониторинга состояния соединения локальной виртуальной сети VLAN 5, имеющей полносвязную топологию, поэтому компьютеры C1, C2 и C3 могут взаимодействовать между собой по принципу «каждый с каждым». Для мониторинга сети VLAN5 созданы точки MEP1, MEP2 и MEP3, располагающиеся на интерфейсах коммутаторов S1, S2 и S5 соответственно.

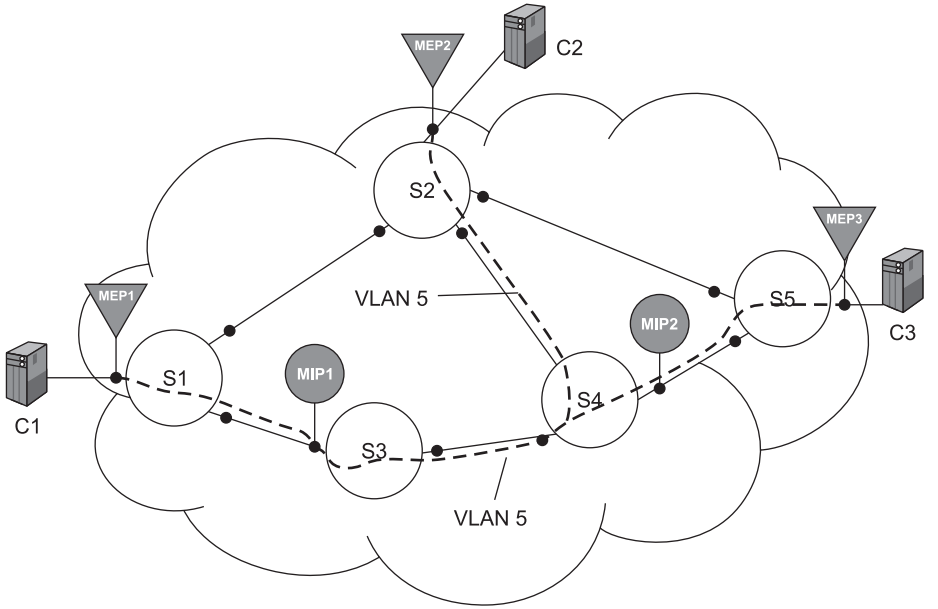


Рис. 12.1. Мониторинг состояния VLAN с помощью протокола CFM

Чтобы осуществлять мониторинг соединений полносвязной топологии, присущей VLAN 5, сообщения CCM посылаются с групповым MAC-адресом. Для мониторинга двухточечных соединений могут использоваться как индивидуальные, так и групповые MAC-адреса. В нашем примере точка MEP1 периодически посылает в сеть VLAN 5 сообщения CCM с групповым адресом. Если сеть VLAN 5 работоспособна, то точки MEP2 и MEP3 регулярно получают сообщения CCM, отправляемые точкой MEP1. Также регулярно передаются и принимаются сообщения CCM, генерируемые точками MEP2 и MEP3. В результате протокол CFM определяет статус сети VLAN 5 как полностью работоспособной.

Предположим теперь, что в сети произошел отказ физического соединения между коммутаторами S4 и S5. Вследствие этого точка MEP3 перестает принимать сообщения CCM от точек MEP1 и MEP2, а они, в свою очередь, — сообщения CCM от точки MEP3. В то же время точки MEP1 и MEP2 по-прежнему продолжают обмениваться сообщениями CCM. Результатом мониторинга будет переход соединения VLAN 5 в состояние частичной работоспособности, когда только часть узлов оказывается достижимой.

Покажем теперь полезность наличия MIP в сети. Предположим, что в сети, показанной на рис. 12.1, связь между коммутаторами S4 и S5 восстановлена, но по какой-то причине потеряна связь между коммутаторами S3 и S4. Точка MEP1 при этом перестает принимать сообщения от точки MEP3, а точка MEP3 — от точки MEP1 (для упрощения анализа мы сейчас игнорируем точку MEP2 и контролируемую ею часть сети). Ясно, что точки MEP1 и MEP3 фиксируют нарушение связности между собой, но их информация не позволяет судить о том, где конкретно в сети возникла проблема — отказ мог произойти в любом из трех сегментов сети между коммутаторами S1 и S5. Но так как в сети имеются точки MIP, администратор может проанализировать их статистику. Статистика MIP1 покажет, что через эту точку по-прежнему проходят сообщения CCM от MEP1, но не проходят сообщения MEP3. Статистика MIP2 покажет обратную картину — наличие сообщений от MEP3, но не от MEP1. Эти данные свидетельствуют о том, что связь потеряна между коммутаторами S3 и S4.

Весьма важной является способность протокола CFM работать в *многодоменной среде*, когда соединение проходит через несколько сетей, принадлежащих разным административным доменам. Каждый из администраторов домена нуждается в мониторинге соединения, но только в пределах своей сети.

Для поддержки мониторинга состояния соединений в многодоменной сети для протокола CFM конфигурируется отдельный *домен мониторинга*, при этом домены мониторинга образуют иерархию доменов различного *уровня мониторинга*. В каждом домене создаются точки обслуживания MEP и MIP, но точки каждого домена работают только с сообщениями CCM своего уровня, а сообщения более высоких уровней просто прозрачно передают. Эту идею иллюстрирует рис. 12.2. Здесь показана сеть, состоящая из доменов различных типов: домена пользователя, домена провайдера услуги виртуальной частной сети и домена оператора связи, через который работает сеть провайдера услуги. В сети имеется три домена операторов: оператора A, оператора B и оператора C. Домены операторов вложены в домен провайдера услуг, который предоставляет пользователю услуги виртуальной локальной сети «из конца в конец». И наконец, на верхнем уровне находится домен пользователя, в который входит домен провайдера услуги.

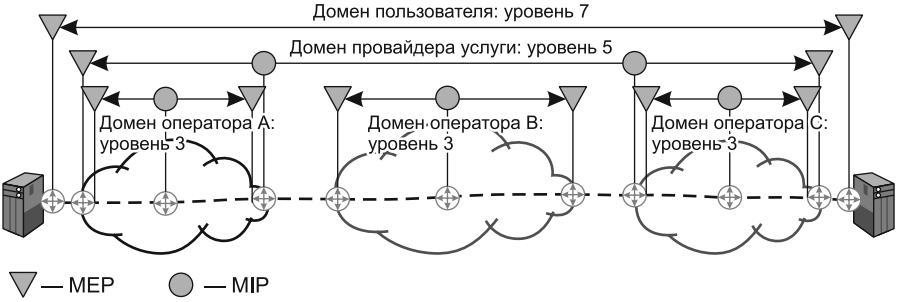


Рис. 12.2. Многодоменное применение протокола CFM

Домену пользователя присвоен уровень 7, домену провайдера — уровень 5, домену оператора связи — уровень 3. Точки MEP на интерфейсах оборудования оператора связи рабо-

тают с сообщениями CCM уровня 3, а сообщения точек обслуживания сети пользователя уровня 7 и сети провайдера услуги уровня 5 они передают прозрачно. Аналогично, точки провайдера услуги работают на интерфейсах его оборудования на уровне 5 и прозрачно передают сообщения CCM уровня домена пользователя 7. Сообщения уровня 3 до точек MEP уровня 5 не доходят, так как завершаются в точках MEP уровня 3.

В результате каждый оператор связи получает информацию о состоянии соединения в пределах своей сети, провайдер — в пределах своей, а пользователь соединения — «из конца в конец». Такой иерархический способ организации сеансов между точками MEP дает возможность проводить независимый мониторинг одного и того же соединения различными организациями без необходимости координировать конфигурацию точек мониторинга — достаточно согласовать используемые каждой организацией уровни точек MEP. Обязательным условием является вложенность доменов каждого уровня в домен более высокого уровня иерархии.

Протокол мониторинга качества соединений Y.1731

Стандарт Y.1731, разработанный ITU-T, добавляет к стандарту CFM возможность измерять следующие дополнительные параметры между точками мониторинга сети:

- *Односторонняя задержка кадра.* Для ее измерения точки обслуживания сети MEP генерируют сообщения измерения задержки и ответа на измерение задержки, в которых переносятся временны́е отметки, позволяющие измерить задержку.
- *Вариация задержки.* Эта задержка измеряется на основе тех же сообщений, что и односторонняя задержка.
- *Потери кадров.* Для измерения этой величины служат сообщения измерения потерь и ответа на измерение потерь. Счетчики сообщений двух точек обслуживания сравниваются, и на основе этого сравнения рассчитываются потери кадров в каждом из направлений.

Стандарт тестирования физического соединения Ethernet

Стандарт тестирования физического соединения Ethernet IEEE 802.3ah предназначен для обнаружения ошибок соединения между двумя непосредственно физически связанными интерфейсами Ethernet. Он поддерживает такие функции, как удаленное обнаружение неисправностей и удаленный контроль обратной связи. Последняя функция является наиболее интересной для специалистов, занимающихся эксплуатацией сетей Ethernet, поскольку позволяет удаленно (через сеть) выдать запрос некоторому интерфейсу Ethernet на переход в режим обратной связи. В данном режиме все кадры, посылаемые на этот интерфейс соседом по линии связи, возвращаются им обратно. Полученные кадры затем можно проанализировать, чтобы установить качество физической линии.

Необходимо отметить, что процедура тестирования линии в режиме обратной связи нарушает нормальную работу соединения, поэтому тестирование нужно проводить в специальное время, отведенное для обслуживания сети.

Интерфейс локального управления Ethernet

Стандарт E-LMI позволяет пограничному пользовательскому устройству, то есть устройству типа CE, запрашивать информацию о состоянии и параметрах услуги, предоставляемой сетью провайдера по данному интерфейсу. Например, пограничный коммутатор Ethernet, расположенный в сети пользователя, может запросить у пограничного коммутатора провайдера информацию о состоянии услуги (работоспособности соединения), предоставляемой по данному интерфейсу. Кроме того, согласно стандарту E-LMI, по запросу можно получить такую информацию об услуге, как отображение идентификатора VLAN пользователя на данное соединение, или же величину пропускной способности, гарантированной для данного соединения.

Мосты провайдера

Стандарт IEEE 802.1 ad на **мосты провайдера** (Provider Bridge, PB) был первым стандартом, который решал проблему изоляции адресного пространства сети провайдера от адресного пространства его пользователей. Этот стандарт был принят IEEE в 2005 году и сегодня реализован в коммутаторах Ethernet многих производителей. Но проблема изоляции адресных пространств решается в этом стандарте лишь частично, так как MAC-адреса пользователей по-прежнему присутствуют в коммутаторах сети провайдера, а разделяются только пространства идентификаторов VLAN.

Стандарт PB вводит двухуровневую иерархию идентификаторов VLAN (рис. 12.3). На внешнем (верхнем) уровне располагается идентификатор VLAN провайдера, называемый **S-VID** (от Service VLAN ID — идентификатор сервиса VLAN), а на нижнем (внутреннем) уровне — идентификатор VLAN пользователя, называемый **C-VID** (от Customer VLAN ID — идентификатор VLAN потребителя).

Идентификатор S-VID помещается в пользовательский кадр пограничным коммутатором провайдера — он проталкивает C-VID в стек и добавляет новый идентификатор S-VID, который потребуется коммутаторам сети провайдера для разделения трафика на виртуальные локальные сети провайдера. Так как S-VID представляет собой новое поле кадра Ethernet, то ему предшествует новое поле типа EtherType, которое на рис. 12.3 обозначено как S-VID-EtherType (в отличие от оригинального поля C-VID-EtherType). Чтобы различать S-VID и C-VID, стандарт 802.1ad вводит новое значение EtherType для типа данных S-VID, равное 0x88a8 (напомним, что для C-VID значение EtherType равно 0x8100). Этот способ инкапсуляции часто неформально называют инкапсуляцией **Q-in-Q**, по названию стандарта 802.1Q, описывающего технику VLAN.

После того как пограничный коммутатор сети провайдера выполняет инкапсуляцию Q-in-Q, кадр обрабатывается магистральными коммутаторами провайдера как обычный кадр, то есть в соответствии с внешним идентификатором VLAN в поле S-VID. Когда кадр прибывает на выходной пограничный коммутатор провайдера, над ним выполняется обратная операция — идентификатор S-VID удаляется. После этого кадр отправляется в сеть пользователя в исходном виде, имея в своем заголовке только идентификатор C-VID.

Внутренние сети VLAN провайдера, соответствующие значениям идентификаторов S-VID, обычно служат для конструирования услуг частных виртуальных сетей. При этом провайдеру нет необходимости согласовывать логическую структуру своей сети с пользователями.

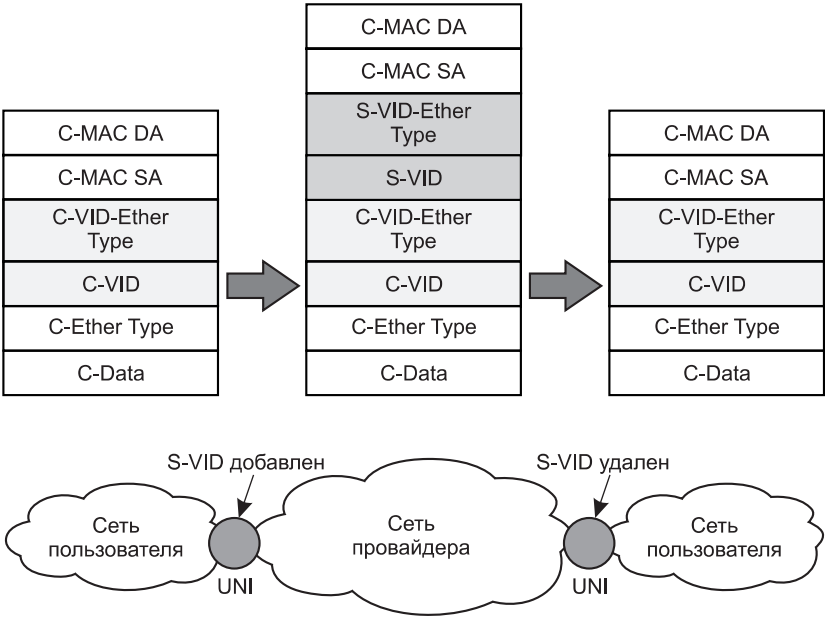


Рис. 12.3. Инкапсуляция идентификаторов VLAN

На рис. 12.4 показана сеть провайдера, которая соединяет **сайты** (так обычно называют сети пользователя при оказании глобальной услуги Ethernet) двух пользователей. Сайты C1, C3 и C5 являются сайтами пользователя *A*, они объединяются в сеть с идентификатором S-VID, равным 156, а сайты C2, C4 и C6 являются сайтами пользователя *B*, они объединяются в сеть с идентификатором S-VID, равным 505.

Конфигурирование услуг сетей 156 и 505 выполнено без учета значений пользовательских идентификаторов VLAN на основании подключения сайта пользователя к некоторому физическому интерфейсу коммутатора провайдера. Так, например, весь пользовательский трафик, поступающий от сайта C1, классифицируется пограничным коммутатором PE1 как принадлежащий виртуальной частной сети с идентификатором S-VID, равным 156. В то же время стандарт PB позволяет провайдеру предоставлять услуги и с учетом значений пользовательских идентификаторов VLAN. Так, если внутри сайта C1 выполнена логическая структуризация и существуют две пользовательские сети VLAN, трафик которых нельзя смешивать, то провайдер может организовать для этого две сети S-VLAN и отображать на них поступающие кадры в зависимости от значений C-VID. Но при своей очевидной полезности стандарт PB имеет несколько недостатков:

- Коммутаторы сети провайдера, как пограничные, так и магистральные, должны изучать MAC-адреса узлов сетей пользователей. Это не является масштабируемым решением.
- Максимальное количество услуг, предоставляемых провайдером, ограничено числом 4096 (так как поле S-VID имеет стандартный размер 12 бит).
- Инжиниринг трафика ограничен возможностями протокола покрывающего дерева RSTP/MSTP.

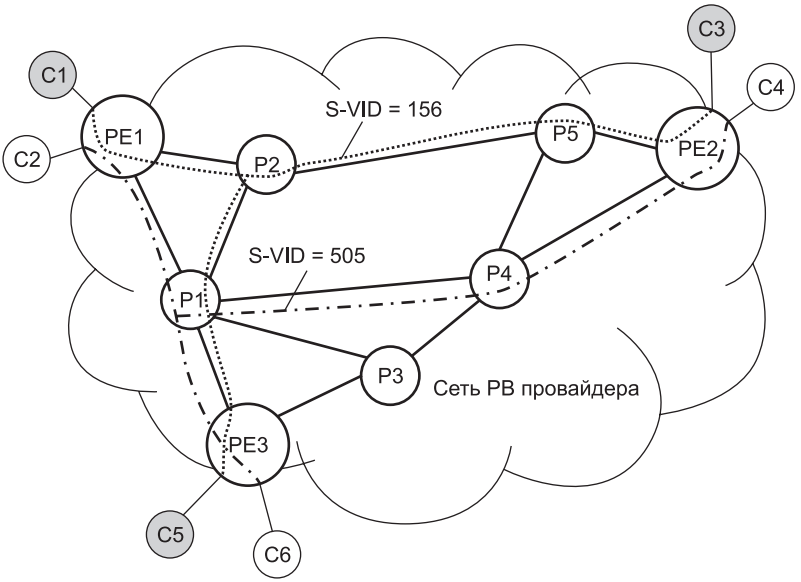


Рис. 12.4. Сеть стандарта PB, предоставляющая услуги соединения сайтов двух пользователей

- Для разграничения деревьев STP, создаваемых в сетях провайдера и пользователей, в стандарте 802.1ad пришлось ввести новый групповой адрес для коммутаторов провайдера. Это обстоятельство не позволяет задействовать в качестве магистральных коммутаторов провайдера те коммутаторы, которые не поддерживают стандарт 802.1ad.

Некоторые из этих недостатков были устранены в стандарте на магистральные мосты провайдера PBB (Provider Bridge Backbone) IEEE 802.1ah, принятом в 2008 году. В этом стандарте адресные пространства пользователей и провайдера разделяются полностью за счет инкапсуляции пользовательских кадров Ethernet в кадры Ethernet сети провайдера. Однако этот стандарт оказался слишком сложным для реализации, и вместо него в крупных сетях провайдеров используются технологии, основанные на MPLS — EVPN (Ethernet VPN) и L2 MPLS VPN (см. главу 21).

С технологией PBB вы можете познакомиться на сайте книги.

(S) *Технология PBB*

Вопросы к части III

1. Верно ли утверждение, что преимуществом разделяемой среды является простота связи между узлами?
2. Верно ли утверждение: «Маркерный доступ обеспечивает менее эффективное использование пропускной способности разделяемой среды»?
3. К какому типу относится MAC-адрес `02:25:86:64:ca:e4`?
4. Преамбула в кадре Ethernet нужна для (выберите вариант ответа):
   а) вхождения приемника в синхронизм;
   б) вхождения передатчика в синхронизм;
   в) расширения кадра для более устойчивого распознавания коллизий.
5. При повышении скорости передачи данных максимальный диаметр сети Ethernet на разделяемой среде (выберите вариант ответа):
   а) увеличивается;
   б) уменьшается;
   в) остается неизменным.
6. Для повышения производительности сети Ethernet количество ее сегментов (при сохранении числа станций в сети) нужно (выберите вариант ответа):
   а) увеличить;
   б) уменьшить.
7. Таблица продвижения моста имеет следующий вид:

MAC-адрес	Порт
`f8:f2:1e:0c:3a:b8`	1
`d0:94:66:4c:37:bf`	2

   На порт 2 коммутатора поступает кадр с адресом назначения `ac:1f:6b:64:c7:d6` и адресом источника `f8:f2:1e:0c:3a:b8`. Укажите, какое действие выполнит мост:
   а) отбросит кадр;
   б) передаст его на все порты и заменит первую запись таблицы продвижения на MAC-адрес `f8:f2:1e:0c:3a:b8`, Порт 2;
   в) передаст его на все порты и не станет корректировать записи.
8. В сети на коммутаторах образовалась петля. Укажите, с помощью каких мер можно предотвратить негативные последствия такой конфигурации:
   а) установить более скоростные порты Ethernet;
   б) активировать протокол STP;
   в) разорвать петлю физическим отсоединением порта;
   г) разорвать петлю логическим переводом порта в неактивное состояние.

9. Верно ли утверждение, что коммутаторы Bare Metal продаются с установленной ОС?
10. Поясните, для чего в стандарте Gigabit Ethernet введено поле расширения:
    а) для увеличения доли пользовательских данных в кадре;
    б) для обеспечения диаметра сети 200 м при работе на разделяемой среде.
11. Существует ли вариант стандарта 10G Ethernet на витой паре?
12. Поясните, за счет чего повышена скорость в стандарте 100GBase-SR4 по сравнению со стандартом 10GBase-SR (выберите вариант ответа):
    а) за счет применения метода кодирования 16-QAM;
    б) повышения тактовой частоты в 4 раза;
    в) повышения тактовой частоты в 2,5 раза;
    г) применения четырех параллельных потоков.
13. Поясните, какие механизмы позволили достичь требуемой скорости в стандарте 400G Ethernet (выберите вариант ответа):
    а) кодирование PAM4;
    б) кодирование PAM5;
    в) применение параллельных волн;
    г) применение кодов FEC.
14. Каким образом можно назначить некоторый коммутатор корневым?
15. Верно ли утверждение: «Порт, имеющий минимальное расстояние до корневого коммутатора среди всех портов данного коммутатора, является назначенным»?
16. Протокол RSTP находит покрывающее дерево быстрее протокола STP, так как он (выберите вариант ответа):
    а) исключает из рассмотрения тупиковые порты;
    б) принимает во внимание существующие VLAN;
    в) назначает резервные порты для корневых и назначенных портов;
    г) сокращает период фиксации отказа в сети.
17. Составьте список доступа, запрещающий узлу с MAC-адресом `f8:f2:1e:0c:3a:b8` обращаться к узлу с MAC-адресом `ac:1f:6b:64:c7:d6`, при этом все остальные взаимодействия в сети должны быть разрешены.
18. Сервер и клиентский компьютер подключены к портам одного и того же коммутатора: сервер — к порту Ethernet10G, а клиент — к Ethernet 1G. На клиенте работает только одно приложение, которое обращается к серверу. Повысится ли скорость обмена клиента с сервером, если клиенту добавить еще один порт Ethernet 1G, соединяющий его с коммутатором, и образовать группу LAG из двух его портов Ethernet 1G?
19. Каким образом можно предотвратить включение некоторого порта в группу LAG протоколом LACP?
20. Поясните, каким образом можно объединить несколько VLAN (выберите вариант ответа):
    а) с помощью маршрутизатора;
    б) приписать их к одному транку.

21. Какое действие выполняет порт доступа, передавая помеченный кадр клиентскому узлу (кадр получен от порта- транка по внутренней шине коммутатора)?
22. Для превращения Ethernet в технологию операторского класса были произведены изменения в нескольких областях, среди них (выберите вариант ответа):
    а) отменена широковещательная рассылка;
    б) разделены адресные пространства пользователя и провайдера;
    в) добавлены функции OAM.
23. Каждая точка MEP выполняет следующие функции (выберите вариант ответа):
    а) генерирует сообщения CCM;
    б) анализирует транзитные IP-пакеты;
    в) принимает сообщения CCM.
24. Корректно ли при конфигурировании протокола CFM назначить домену оператора уровень 6, а домену пользователя — уровень 3?

Часть IV

Сети TCP/IP

- ❑ Глава 13. Стек протоколов TCP/IP. Адресация
- ❑ Глава 14. Протокол межсетевого взаимодействия IP
- ❑ Глава 15. Протоколы транспортного уровня TCP и UDP
- ❑ Глава 16. Протоколы маршрутизации и технология SDN
- ❑ Глава 17. IPv6 как развитие стека TCP/IP

Эта часть книги посвящена самой популярной сетевой технологии, появившейся более 50 лет назад в результате создания интернета и использующейся сегодня практически во всех существующих и вновь создаваемых сетях.

Следуя логике, диктуемой моделью OSI, вслед за частями II и III, в которых рассматривались технологии физического и канального уровней, в этой части книги мы обратимся к средствам сетевого уровня, то есть таким, которые обеспечивают возможность объединения множества разных сетей в единую сеть. Учитывая, что бесспорным лидером среди протоколов сетевого уровня является протокол IP, мы будем рассматривать вопросы построения объединенных сетей на его примере. При этом мы постараемся дать широкую картину взаимодействия всех протоколов стека TCP/IP Следуя важной современной тенденции все более широкого использования протокола IPv6, мы уделим внимание принятым в этой версии системе адресации и маршрутизации, а также тем возможностям, которые предлагаются для плавного перехода на эту новую версию.

Забегая вперед, хотим предупредить читателя, что в следующих частях книги мы еще не раз обратимся к стеку TCP/IP. Так, в части V «Глобальные проводные коммуникационные сети» изучаются прикладные протоколы стека TCP/IP: HTTP (веб-служба), SMTP, IMAP и POP (почтовая служба), SNMP и telnet (системы управления сетью), а также тесно связанная с IP коммуникационная технология MPLS. В части VI «Беспроводная передача данных» вы познакомитесь с той ролью, которую играет протокол IP в работе беспроводных сетей.

Материал последней части VII «Безопасность компьютерных сетей» также тесно связан с протоколами и технологиями IP-сетей. В ней рассматриваются уязвимости и методы защиты транспортных протоколов и информационных служб IP-сетей, фильтрация IP-трафика, протокол трансляции IP-адресов NAT, защищенная версия протокола IP— протокол IPSec.

ГЛАВА 13 Стек протоколов TCP/IP. Адресация

Стек протоколов TCP/IP

Мы уже обсуждали коротко уникальность стека протоколов TCP/IP в разделе «Стандартные стеки коммуникационных протоколов» главы 4. В следующих 5 главах мы рассмотрим особенности технологии TCP/IP более подробно.

На рис. 13.1 показано каноническое представление стека TCP/IP. Соответствие стека TCP/IP рекомендациям модели OSI весьма условно. Разработчики стека отдали предпочтение скорости работы сети в ущерб модульности. Вместо 7 уровней в стеке определены 4 уровня: прикладной, транспортный, сетевой и уровень сетевых интерфейсов.

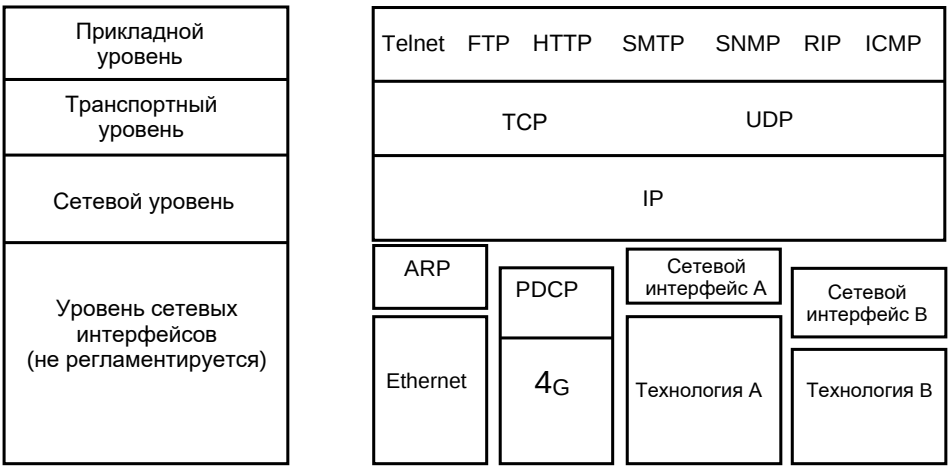


Рис. 13.1. Структура стека TCP/IP

Прикладной уровень стека TCP/IP вбирает в себя функции сеансового уровня, уровня представления и прикладного уровня модели OSI. Он объединяет сервисы, предоставляемые стеком TCP/IP пользовательским приложениям. К ним относятся протокол передачи файлов (File Transfer Protocol, FTP), протокол эмуляции терминала telnet, простой протокол передачи почты (Simple Mail Transfer Protocol, SMTP), протокол передачи гипертекста (Hypertext Transfer Protocol, HTTP) и многие другие. На рисунке к прикладному уровню отнесены также протоколы, выполняющие функции, вспомогательные по отношению к задаче передачи данных, а именно: протоколы RIP и OSPF, предназначенные для определения маршрутов и составления таблиц маршрутизации, протокол межсетевых управляющих

сообщений (Internet Control Message Protocol, ICMP), предназначенный для передачи источнику сведений об ошибках, возникших при передаче пакета, и протоколы управления сетью SNMP, а также некоторые другие протоколы. (Некоторые из этих протоколов часто относят к сетевому уровню, поскольку задачи, которые они выполняют, близки к задачам протокола IP.) Протоколы прикладного уровня работают на *хостах*[1].

Транспортный уровень стека TCP/IP может предоставлять вышележащему уровню два типа сервиса:

- гарантированную доставку обеспечивает **протокол управления передачей** (Transmission Control Protocol, **TCP**);
- доставку по возможности, или с максимальными усилиями, обеспечивает **протокол пользовательских дейтаграмм** (User Datagram Protocol, **UDP**).

Чтобы обеспечить надежную доставку данных, протокол TCP предусматривает установление *логического соединения*. Это позволяет нумеровать пакеты, подтверждать их прием квитанциями, организовывать в случае потери повторные передачи, распознавать и уничтожать дубликаты, доставлять прикладному уровню пакеты в том порядке, в котором они были отправлены. Благодаря этому протоколу объекты на хосте-отправителе и хосте-получателе могут поддерживать обмен данными в дуплексном режиме. TCP дает возможность без ошибок доставить сформированный на одном из компьютеров поток байтов на любой другой компьютер, входящий в составную сеть.

Второй протокол этого уровня, UDP, является простейшим *дейтаграммным* протоколом, используемым, если задача надежного обмена данными либо вообще не ставится, либо решается средствами более высокого уровня — прикладным уровнем или пользовательскими приложениями.

В функции протоколов TCP и UDP входит также исполнение роли *связующего звена* между прилегающими к транспортному уровню прикладным и сетевым уровнями. От прикладного протокола транспортный уровень принимает задание на передачу данных с тем или иным качеством прикладному уровню-получателю. Нижележащий сетевой уровень протоколы TCP и UDP рассматривают как своего рода инструмент, пусть и не очень надежный, но способный перемещать пакет в свободном и рискованном путешествии по составной сети. Программные модули, реализующие протоколы TCP и UDP, подобно модулям протоколов прикладного уровня, устанавливаются на *хостах*.

Сетевой уровень, называемый также **уровнем интернета**, является стержнем всей архитектуры TCP/IP.

Основным протоколом сетевого уровня является межсетевой протокол **IP** (Internet Protocol). В его задачу входит продвижение пакетов между сетями — от одного маршрутизатора к другому до тех пор, пока пакеты не попадут в сеть назначения. Протокол IP поддерживает интерфейс с вышележащим *транспортным уровнем*, получая от него запросы на передачу данных по составной сети, а также с нижележащим *уровнем сетевых интерфейсов*, о функциях которого мы расскажем далее. В отличие от протоколов прикладного и транспортного уровней, протокол IP устанавливается не только на хостах, но

---

[1] В Интернете (а значит, и в стеке протоколов TCP/IP) конечный узел традиционно называют *хостом*, а маршрутизатор — *шлюзом*. Далее мы будем использовать пары терминов «конечный узел» — «хост» и «маршрутизатор» — «шлюз» как синонимы, чтобы отдать дань уважения традиционной терминологии Интернета и в то же время не отказываться от современных терминов.

и на всех *маршрутизаторах*. Протокол IP — это дейтаграммный протокол, работающий без установления соединений по принципу доставки с максимальными усилиями (best effort). Такой тип сетевого сервиса называют также «ненадежным».

Идеологическим отличием архитектуры стека TCP/IP от многоуровневой архитектуры других стеков является интерпретация функций самого нижнего уровня, названного разработчиками канальным (link layer). Чтобы отразить функциональное отличие нижнего уровня стека TCP/IP от канального уровня модели OSI (data link layer), чаще используют другое название этого уровня — **уровень сетевых интерфейсов** (network interface layer).

Напомним, модель OSI регламентирует набор функций, которые должны выполняться на канальном и физическом уровнях: доступ к среде передачи, установление логических соединений, формирование кадров, контроль корректности передачи и повторную передачу искаженных и потерянных кадров, согласование величин электрических сигналов, кодирование, синхронизация, усиление. Конкретные реализации этих функций составляют суть протоколов физического и канального уровней, таких, например, как Ethernet и PPP.

В отличие от модели OSI стек TCP/IP ожидает от нижележащей технологии выполнение *единственной функции — доставки данных между двумя IP-узлами*. Протокол IP не интересует, будет ли нижележащая технология передавать его пакет с установлением логического соединения или нет, выполняет ли она повторные передачи ошибочных и утерянных кадров, какую передающую среду — проводную или беспроводную — она использует. Если же качество передачи пакетов IP нижележащей технологией не устраивает стек TCP/IP, то он повышает это качество собственными средствами. Например, если часть данных, передаваемых пакетами IP, оказывается потеряна, то ее повторную передачу осуществляет протокол TCP.

Таким образом, в стеке TCP/IP функции канального и физического уровня *не регламентируются*.

Поскольку набор услуг, который сетевой протокол IP требует от нижележащих технологий, *минимален* (доставка данных между двумя узлами сети), то это упрощает интерфейс «протокол IP — технология нижележащего уровня» и вследствие этого упрощается адаптация новых технологий в стек протоколов TCP/IP. Этот интерфейс служит для решения двух задач:

- *упаковка* (*инкапсуляция*) IP-пакета в единицу передаваемых данных промежуточной сети;
- *отображение сетевых IP-адресов* в адреса технологии данной промежуточной сети.

Инкапсуляция IP-пакета в кадр нижележащей технологии может потребовать использования согласующего протокола, если максимальный размер кадра оказывается недостаточным для помещения в его поле данных IP-пакета максимальной длины. Такие протоколы-прокладки были разработаны для некоторых популярных технологий, например, для технологий сотовой связи 4G и 5G.

Отображение IP-адресов на адреса технологий сетевых интерфейсов может выполняться автоматически некоторым протоколом или же вручную администратором сети. Примером протокола автоматического отображения адресов является протокол ARP, принадлежащий стеку TCP/IP. Он использует широковещание сетей Ethernet для нахождения MAC-адреса узла Ethernet, имеющего определенный IP-адрес.

Любая коммуникационная среда, которая позволяет двум или более IP-узлам непосредственно взаимодействовать друг с другом без промежуточных маршрутизаторов, рассматривается протоколом IP как одна **линия связи (Link)**, как если бы эти узлы были связаны отрезком кабеля.

В простейшем случае такая коммуникационная среда может действительно представлять собой отдельную физическую линию связи с работающим на ней протоколом канального уровня, например Ethernet или PPP. Более сложным случаем является локальная сеть, построенная на коммутаторах. Но поскольку она также является доменом широковещания, то для протокола IP эта коммутируемая среда неотличима от отдельной физической линии связи. Такой гибкий подход упрощает решение проблемы расширения набора поддерживаемых технологий. При появлении новой популярной технологии она быстро включается в стек TCP/IP путем разработки соответствующего стандарта, определяющего метод инкапсуляции IP-пакетов в ее кадры и протокола отображения адресов.

Каждый коммуникационный протокол оперирует некоторой единицей передаваемых данных. Названия этих единиц иногда закрепляются стандартом, а чаще просто определяются традицией. В стеке TCP/IP за многие годы его существования образовалась устоявшаяся терминология в этой области (рис. 13.2).

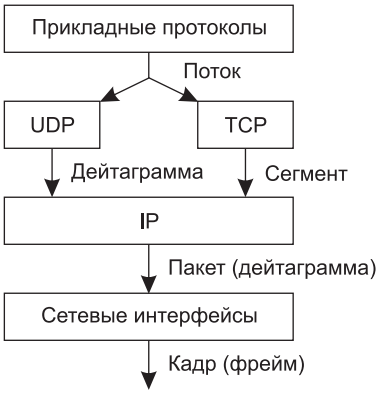


Рис. 13.2. Названия протокольных единиц данных в TCP/IP

Потоком данных, информационным потоком или просто **потоком** называют данные, поступающие от приложений на вход протоколов транспортного уровня — TCP и UDP. Протокол TCP «нарезает» из потока данных **сегменты**. Единицу данных протокола UDP часто называют **дейтаграммой**. Дейтаграмма — это общее название для единиц данных, которыми оперируют протоколы *без установления соединений*. К таким протоколам относится и протокол IP, поэтому его единицу данных иногда тоже называют дейтаграммой, хотя чаще используется другой термин — **пакет**.

В стеке TCP/IP единицы данных любых технологий, в которые упаковываются IP-пакеты для последующей передачи через сети составной сети, принято называть **кадрами** или **фреймами**. При этом не имеет значения, какое название используется для этой единицы данных в технологии составляющей сети. Для TCP/IP фреймом является и кадр Ethernet и кадр Wi-Fi, в тех случаях, когда они выступают в качестве контейнера, в котором IP-пакет переносится через составную сеть.

Межсетевое взаимодействие

Протокол IP служит для образования единой транспортной системы, объединяющей несколько сетей и называемой **составной сетью**.

Технология, позволяющая соединять в единую сеть множество сетей, в общем случае построенных на основе разных технологий, называется **технологией межсетевого взаимодействия** (internet working).

На рис. 13.3 показано несколько сетей, каждая из которых использует собственную технологию продвижения данных: Ethernet, 4G, MPLS/PPP и Wi-Fi. На базе этих технологий любая из указанных сетей может связывать между собой любых пользователей, но только в пределах *своей* сети, и не способна обеспечить передачу данных в другую сеть. Причина такого положения вещей очевидна и кроется в существенных отличиях одной технологии от другой.

Для связывания между собой сетей, построенных на основе отличающихся технологий, используются специальные коммутационные узлы сети — маршрутизаторы с установленными на них протоколом IP и протоколами маршрутизации.

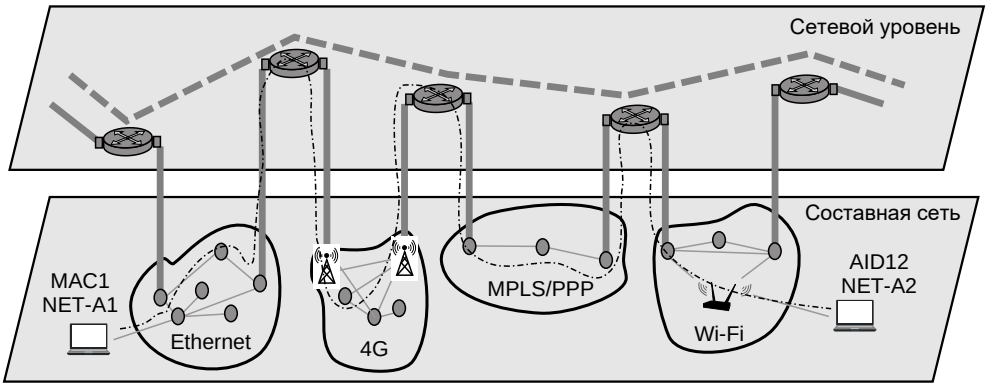


Рис. 13.3. Передача данных через составную сеть

Одной из функций маршрутизатора является *физическое соединение сетей*. Маршрутизатор имеет несколько сетевых интерфейсов, подобных интерфейсам компьютера, к каждому из которых может быть подключена одна сеть. Таким образом, *интерфейсы маршрутизатора можно считать узлами разных сетей*. Это означает, что каждый интерфейс маршрутизатора поддерживает протоколы физического и канального уровней той сети, с которой он соединен, то есть интерфейсы Ethernet, 4G и т. д.

Рассмотрим, как маршрутизаторы *передают данные через составную сеть между двумя конечными узлами A1 и A2*. Данные поступают на сетевой уровень компьютера A1 от одного из транспортных протоколов TCP или UDP этого узла. Они снабжаются заголовком сетевого уровня. Данные вместе с заголовком образуют **пакет**. В заголовке пакета наряду с другой служебной информацией содержится адрес назначения этого пакета — в данном случае адрес компьютера A2.

Чтобы протоколы сетевого уровня могли доставлять пакеты любому узлу составной сети, эти узлы должны наряду с адресами, назначенными им на канальном уровне (например,

MAC-адресами), иметь адреса, *уникальные в пределах всей составной сети*, называемые **сетевыми адресами**. Например, на рис. 13.3 компьютер в сети Ethernet имеет адрес канального уровня MAC1 и адрес сетевого уровня NET-A1; аналогично в сети Wi-Fi узел, адресуемый локальным идентификатором AID12, имеет сетевой адрес NET-A2.

Для продвижения пакетов маршрутизатор должен знать маршрут. На сетевом уровне маршрут описывается последовательностью *сетей* (или маршрутизаторов), через которые должен пройти пакет, чтобы попасть к адресату. Для задания маршрута в каждом маршрутизаторе формируется таблица маршрутизации, называемая также таблицей продвижения. Таблица маршрутизации может быть получена разными способами, описанию которых посвящена глава 16 «Протоколы маршрутизации и технология SDN». Сегодня основным подходом к построению таблиц маршрутизации является согласованная работа маршрутизаторов, в результате которой каждый маршрутизатор строит собственную таблицу маршрутизации на основе обмена маршрутной информацией с остальными маршрутизаторами сети.

В соответствии с многоуровневым подходом сетевой уровень для решения своей задачи обращается к услугам нижележащего, канального уровня. Весь путь через составную сеть разбивается на участки от одного маршрутизатора до другого, причем каждый участок соответствует пути через отдельную сеть.

Чтобы передать пакет через очередную сеть, сетевой уровень помещает его в поле данных кадра соответствующей канальной технологии. В заголовке кадра указывается *адрес канального уровня* интерфейса следующего маршрутизатора. Сеть, используя свою канальную технологию, доставляет кадр с инкапсулированным в него пакетом по заданному адресу. Маршрутизатор извлекает пакет из прибывшего кадра и после необходимой обработки передает пакет для дальнейшей транспортировки в следующую сеть, предварительно упаковав его в новый кадр канального уровня в общем случае другой технологии. Таким образом, межсетевой протокол IP играет роль *координатора*, организующего совместную работу сетей, построенных на основе разных технологий.

Для маршрутизаторов сети путь пакета через промежуточную сеть прозрачен, они его не «замечают», для них это соединение «точка-точка», что показано в верхний части рисунка пунктирной линией.

Типы адресов стека TCP/IP

Одним из достоинств технологии TCP/IP является *гибкость и масштабируемость системы адресации*, что позволяет ей достаточно просто включать в составную сеть сети разных технологий и разного масштаба. Основными задачами адресации являются следующие:

- *Согласованное использование адресов различного типа.* Эта задача включает отображение адресов разных типов друг на друга, например сетевого IP-адреса на локальный, доменного имени — на IP-адрес.
- *Обеспечение уникальности адресов.* В зависимости от типа адреса требуется обеспечивать однозначность адресации в пределах компьютера, подсети, корпоративной сети или интернета.
- *Конфигурирование сетевых интерфейсов и сетевых приложений.*

Каждая из перечисленных задач имеет достаточно простое решение для сети, число узлов которой не превосходит нескольких десятков. Например, для отображения символьного доменного имени на IP-адрес достаточно поддерживать на каждом хосте таблицу всех символьных имен, используемых в сети, и соответствующих им IP-адресов. Столь же просто «вручную» присвоить всем интерфейсам в небольшой сети уникальные адреса. Однако в крупных сетях эти же задачи усложняются настолько, что требуют принципиально иных решений.

Ключевым словом, которое характеризует принятый в TCP/IP подход к решению этих проблем, является **масштабируемость**. Процедуры, предлагаемые TCP/IP для назначения, отображения и конфигурирования адресов, одинаково хорошо работают в сетях разного масштаба. В этой главе, наряду с собственно схемой образования IP-адресов, мы познакомимся с наиболее популярными масштабируемыми средствами поддержки адресации в сетях TCP/IP: технологией бесклассовой междоменной маршрутизации, системой доменных имен и протоколом динамического конфигурирования хостов.

> В этой главе речь идет о системе адресации, используемой в четвертой версии протокола **IPv4**, которая существенно отличается от системы адресации в версии **IPv6**. Особенности IPv6 рассматриваются в главе 17.

Для идентификации сетевых интерфейсов используются три типа адресов:

- локальные (аппаратные) адреса;
- сетевые адреса (IP-адреса);
- символьные (доменные) имена.

В разных сетевых технологиях в общем случае используются собственные системы адресации, предназначенные исключительно для обеспечения связи собственных узлов. Но как только некоторая сеть объединяется с другими сетями в составную, функциональность этих адресов расширяется, и они становятся необходимым элементом вышележащей объединяющей технологии — в данном случае TCP/IP. Роль, которую играют эти адреса в TCP/IP, не зависит от того, какая именно технология применяется в подсети, поэтому они имеют общее название — **локальные (аппаратные) адреса**. Например, если в составную сеть включена подсеть Ethernet, то локальными адресами сетевых интерфейсов этой сети для технологии TCP/IP будут соответственно MAC-адреса, а если подсеть ATM — то номера виртуальных каналов.

ПРИМЕЧАНИЕ

Слово «локальный» в контексте TCP/IP означает «действующий не во всей составной сети, а лишь в пределах подсети». Именно в таком смысле понимаются здесь термины: «локальная технология» (технология, на основе которой построена подсеть), «локальный адрес» (адрес, применяемый некоторой локальной технологией для адресации узлов в пределах подсети). Напомним, что в качестве подсети («локальной сети») может выступать и LAN, и WAN. Следовательно, говоря о подсети, мы задействуем слово «локальная» не как характеристику технологии, лежащей в основе этой подсети, а как указание на роль, которую играет эта подсеть в архитектуре составной сети. Сложности могут возникнуть и при интерпретации определения «аппаратный». В данном случае термин «аппаратный» подчеркивает концептуальное представление разработчиков стека TCP/IP о подсети как о некотором вспомогательном аппаратном средстве, единственной функцией которого является перемещение IP-пакета через подсеть до ближайшего шлюза.

Чтобы технология TCP/IP могла решать свою задачу объединения сетей, ей необходима *собственная глобальная система адресации,* позволяющая универсальным и однозначным способом идентифицировать любой интерфейс составной сети. Очевидным решением является нумерация всех подсетей составной сети, а затем нумерация сетевых интерфейсов в пределах каждой из этих подсетей. Пара, состоящая из номера сети и номера узла[1], отвечает поставленным условиям и может служить в качестве **сетевого адреса**, или **IP-адреса**. Сетевой адрес представляет собой набор чисел, например 192.45.66.17.

Числовое представление сетевого адреса достаточно эффективно для программных и аппаратных средств. Однако пользователи обычно предпочитают работать с более удобными **символьными именами** компьютеров. Именно с символьными именами вы имеете дело, когда задаете веб-браузеру адрес доступа к тому или иному сайту интернета. Символьные имена в пределах составной сети строятся по иерархическому признаку. Примером доменного имени может служить имя **base2.sales.zil.ru**. Символьные имена называют также **доменными** или **DNS-именами**.

> Между локальным адресом, доменным именем и IP-адресом, относящимся к одному и тому же сетевому интерфейсу, нет функциональной зависимости, поэтому единственный путь получить отображение адреса одного типа в адрес другого типа — это построить *таблицу соответствия*.

В общем случае сетевой интерфейс может иметь несколько локальных адресов, сетевых адресов и доменных имен.

Формат IP-адреса

В заголовке IP-пакета предусмотрены поля для хранения *IP-адреса отправителя* и *IP-адреса получателя*. Каждое из этих полей имеет фиксированную длину 4 байта (32 бита). Как уже было сказано, IP-адрес состоит из двух логических частей — номера сети и номера узла в сети. Наиболее распространенная форма представления IP-адреса — запись в виде четырех чисел, представляющих значения каждого байта в десятичной форме и разделенных точками, например:

128.10.2.30

Этот же адрес может быть представлен в двоичном формате:

10000000 00001010 00000010 00011110

В шестнадцатеричном формате тот же адрес имеет следующий вид:

80.0A.02.1D

Запись адреса не предусматривает *специального разграничительного знака* между номером сети и номером узла. Вместе с тем при передаче пакета по сети часто возникает необходимость разделить адрес на эти две части. Например, маршрутизация, как правило, осуществляется на основании номера сети, поэтому каждый маршрутизатор, получая пакет, должен прочитать из соответствующего поля заголовка адрес назначения и выделить из него номер сети. Каким образом маршрутизаторы определяют, какая часть из 32 бит, отведенных под IP-адрес, относится к номеру сети, а какая — к номеру узла?

---

[1] Напомним, для сокращения часто говорят «узел», а не «сетевой интерфейс узла».

Можно предложить несколько вариантов решения этой проблемы.

Простейший из них состоит в использовании **фиксированной границы**. При этом все 32-битное поле адреса заранее делится на две части не обязательно равной, но фиксированной длины, в одной из которых всегда будет размещаться номер сети, в другой — номер узла. Решение очень простое, но хорошее ли? Поскольку поле, которое отводится для хранения номера узла, имеет фиксированную длину, все сети будут иметь одинаковое максимальное число узлов. Если, например, под номер сети отвести один первый байт, то все адресное пространство распадется на сравнительно небольшое (2^8) число сетей огромного размера (2^{24} узлов). Если границу передвинуть дальше вправо, то сетей станет больше, но все равно все они будут одинакового размера. Очевидно, что такой жесткий подход не позволяет дифференцированно удовлетворять потребности отдельных предприятий и организаций. Именно поэтому он не нашел применения, хотя и использовался на начальном этапе существования технологии TCP/IP.

Второй подход (RFC 950, RFC 1518) основан на применении *маски*, которая позволяет максимально гибко устанавливать границу между номером сети и номером узла. При таком подходе адресное пространство можно использовать для создания множества сетей разного размера.

> **Маска** — это число, применяемое в паре с IP-адресом, причем двоичная запись маски содержит непрерывную последовательность единиц в тех разрядах, которые должны в IP-адресе интерпретироваться как номер сети. Граница между последовательностями единиц и нулей в маске соответствует границе между номером сети и номером узла в IP-адресе.

Например, если маска, связываемая с некоторым IP-адресом, имеет вид 11111111111000 00000000000000000, то номеру сети соответствуют 10 старших разрядов в двоичном представлении данного IP-адреса.

И наконец, способ (RFC 791), основанный на **классах адресов**, представляет собой компромисс по отношению к двум предыдущим: размеры сетей хотя и не могут быть произвольными, как при использовании масок, но и не должны быть одинаковыми, как при установлении фиксированных границ. Вводится пять классов адресов: A, B, C, D, E. Три из них — A, B и C — предназначены для адресации сетей, а два — D и E — имеют специальное назначение. Для каждого класса сетевых адресов определено собственное положение границы между номером сети и номером узла.

Классы IP-адресов

Признаками, на основании которых IP-адрес относят к тому или иному классу, являются значения нескольких первых битов адреса (рис. 13.4).

В табл. 13.1 приведены диапазоны адресов и максимальное число сетей и узлов, соответствующих каждому классу.

Исходя из приведенной структуры адресов и информации из таблицы можно сделать несколько очевидных выводов. Сетей класса A сравнительно немного, зато количество узлов в них очень большое и может достигать 2^{24}, что равно 16 777 216 узлов. Сетей класса B больше, чем сетей класса A, но их размеры меньше, а максимальное количество узлов в сетях класса B составляет 2^{16} (65 536). Сетей класса C больше всего, но они характеризуются самым маленьким максимально возможным количеством узлов, всего 2^8 (256).

Таблица 13.1. Классы IP-адресов

Класс	Первые биты	Наименьший номер сети	Наибольший номер сети	Максимальное число узлов в сети
A	0	1.0.0.0 (0 — не используется)	126.0.0.0 (127 — зарезервирован)	2^{24}, поле 3 байта
B	10	128.0.0.0	191.255.0.0	2^{16}, поле 2 байта
C	110	192.0.0.0	223.255.255.0	2^{8}, поле 1 байт
D	1110	224.0.0.0	239.255.255.255	Групповые адреса
E	11110	240.0.0.0	247.255.255.255	Зарезервировано



Рис. 13.3. Классы IP-адресов

Адреса классов A, B и C используются для идентификации отдельных сетевых интерфейсов, то есть являются **индивидуальными адресами** (unicast address). **Групповые адреса** (multicast address), принадлежащие классу D, не делятся на номер сети и номер узла и обрабатываются маршрутизатором особым образом. Один групповой адрес идентифицирует *группу* сетевых интерфейсов, в общем случае принадлежащих разным сетям. Адрес класса D начинается с последовательности 1110, а в младших адресах содержит *номер (идентификатор) группы*.

Основное назначение групповых адресов — распространение информации по схеме «один ко многим». Интерфейс, входящий в группу, получает наряду с обычным индивидуальным IP-адресом также групповой адрес. Групповые адреса предназначены для экономичного распространения в интернете или большой корпоративной сети аудио- или видеопрограмм, адресованных сразу большой аудитории слушателей или зрителей.

Если групповой адрес помещен в поле адреса назначения IP-пакета, то данный пакет должен быть доставлен сразу нескольким узлам, которые образуют группу с номером, указанным в поле адреса. Один и тот же узел может входить в несколько групп, и узел может в любое время присоединиться либо выйти из группы. Члены группы могут распределяться по различным сетям, находящимся друг от друга на произвольно большом расстоянии.

Номера групп могут быть *временными*, назначаемыми только на время некоторого события (конференции), или *постоянными (well known)*, централизованно закрепленными за типичными группами устройств. Например, адреса 224.0.0.1 и 224.0.0.2 приписаны всем хостам и маршрутизаторам локальной сети соответственно. Так, если хост посылает пакет с адресом назначения 224.0.0.1, то он должен быть доставлен всем хостам в его локальной сети[1]. Если же адрес начинается с последовательности 11110, то это значит, что данный адрес относится к **классу E**. Адреса этого класса зарезервированы для будущих применений.

Чтобы получить из IP-адреса номер сети и номер узла, требуется не только разделить адрес на две соответствующие части, но и дополнить каждую из них нулями до полных четырех байтов. Возьмем, например, адрес класса B 129.64.134.5. Первые два байта идентифицируют сеть, а последующие два — узел. Таким образом, номером сети является адрес 129.64.0.0, а номером узла — адрес 0.0.134.5.

Особые IP-адреса

В TCP/IP существуют ограничения при назначении IP-адресов, а именно: *номера сетей и номера узлов не могут состоять из одних двоичных нулей или единиц.* Отсюда следует, что максимальное количество узлов, приведенное в табл. 13.1 для сетей каждого класса, должно быть уменьшено на 2. Например, в адресах класса C под номер узла отводится 8 бит, которые позволяют задать 256 номеров: от 0 до 255. Однако в действительности максимальное число узлов в сети класса C не может превышать 254, так как адреса 0 и 255 запрещены для адресации сетевых интерфейсов. Из этих же соображений следует, что конечный узел не может иметь адрес типа 98.255.255.255, поскольку номер узла в этом адресе класса A состоит из одних двоичных единиц.

Введя эти ограничения, разработчики технологии TCP/IP получили возможность расширить функциональность системы адресации следующим образом:

- Если IP-адрес состоит только из двоичных нулей, то он называется **неопределенным адресом** и обозначает адрес того узла, который сгенерировал этот пакет. Адрес такого вида в особых случаях помещается в заголовок IP-пакета, в поле адреса отправителя.
- Если в поле номера сети стоят только нули, то по умолчанию считается, что узел назначения принадлежит той же самой сети, что и узел, который отправил пакет. Такой адрес также может быть использован только в качестве адреса отправителя.
- Если все двоичные разряды IP-адреса равны 1, то пакет с таким адресом назначения должен рассылаться всем узлам, находящимся в той же сети, что и источник этого

[1] О групповых адресах читайте также в разделе «Типы адресов IPv6» главы 17 и в разделе « Групповое вещание» главы 16.

пакета. Такой адрес называется **ограниченным широковещательным** (limited broadcast). Ограниченность в данном случае означает, что пакет не выйдет за границы данной подсети ни при каких условиях.

- Если в поле адреса назначения в разрядах, соответствующих номеру узла, стоят только единицы, то пакет, имеющий такой адрес, рассылается всем узлам сети, номер которой указан в адресе назначения. Например, пакет с адресом 192.190.21.255 будет направлен всем узлам сети 192.190.21.0. Такой тип адреса называется **широковещательным** (broadcast).

ВНИМАНИЕ

В протоколе IP нет понятия «широковещание» в том смысле, в котором оно используется в протоколах канального уровня локальных сетей, когда данные должны быть доставлены абсолютно всем узлам сети. Как ограниченный, так и обычный варианты широковещательной рассылки имеют пределы распространения в составной сети: они ограничены либо сетью, которой принадлежит источник пакета, либо сетью, номер которой указан в адресе назначения. Поэтому деление сети с помощью маршрутизаторов на части локализует широковещательный шторм пределами одной из подсетей просто потому, что нет способа адресовать пакет одновременно всем узлам всех сетей составной сети.

Особый смысл имеет IP-адрес, первый октет которого равен 127. Этот адрес является *внутренним адресом стека протоколов* компьютера (или маршрутизатора). Он используется для тестирования программ, а также для организации работы клиентской и серверной частей приложения, установленных на одном компьютере. Обе программные части данного приложения спроектированы в расчете на то, что они будут обмениваться сообщениями по сети. Но какой же IP-адрес они должны использовать для этого? Адрес сетевого интерфейса компьютера, на котором они установлены? Но это приводит к избыточным передачам пакетов в сеть. Экономичным решением является применение внутреннего адреса 127.0.0.0. В IP-сети запрещается присваивать сетевым интерфейсам IP-адреса, начинающиеся со значения 127. Когда программа посылает данные по IP-адресу 127.x.x.x, то данные не передаются в сеть, а возвращаются модулям верхнего уровня этого же компьютера как только что принятые. Маршрут перемещения данных образует «петлю», поэтому этот адрес называется **адресом обратной петли** (loopback).

Использование масок при IP-адресации

Снабжая каждый IP-адрес маской, можно отказаться от понятий классов адресов и сделать систему адресации более гибкой.

Пусть, например, для IP-адреса 129.64.134.5 (0000001.01000000.10000110.00000101) указана маска 255.255.128.0 (11111111.11111111.10000000.00000000).

Если игнорировать маску и интерпретировать адрес 129.64.134.5 на основе классов, то номером сети является 129.64.0.0, а номером узла — 0.0.134.5 (поскольку адрес относится к классу B). Если же использовать маску, то 17 последовательных двоичных единиц в маске 255.255.128.0, «наложенные» на IP-адрес 129.64.134.5, делят его на две части:

- номер сети: 10000001.01000000.1;
- номер узла: 0000110.00000101.

В десятичной форме записи номера сети и узла, дополненные нулями до 32 бит, выглядят соответственно как 129.64.128.0 и 0.0.6.5.

Наложение маски можно интерпретировать как выполнение операции логического умножения, называемой также операцией И (AND). Так, в данном примере номер сети является результатом операции

0000001.01000000.10000110.00000101 AND 11111111.11111111.10000000.00000000.

Помимо *десятичной* и *двоичной* формы представления маски используются и другие форматы. Например, удобно записывать маску в *шестнадцатеричном* коде: FF.FF.00.00 — маска для адресов класса B. Еще чаще встречается запись с *префиксом* 185.23.44.206/26 — данная запись говорит о том, что маска для этого адреса содержит 26 единиц. В табл. 13.2 приведены маски для стандартных классов сетей, записанные в разных форматах.

Таблица 13.2. Маски для стандартных классов сетей

Класс адресов	Десятичная форма	Двоичная форма	Шестнадцатеричная форма	Префикс
A	255.0.0.0	11111111. 00000000. 00000000. 00000000	FF.00.00.00	/8
B	255.255.0.0	11111111. 11111111. 00000000. 00000000	FF.FF.00.00	/16
C	255.255.255.0	11111111. 11111111. 11111111. 00000000	FF.FF.FF.00	/24

Порядок назначения IP-адресов

По определению схема IP-адресации должна обеспечивать уникальность нумерации сетей, а также уникальность нумерации узлов в пределах каждой из сетей. Следовательно, процедуры назначения номеров как сетям, так и узлам сетей должны быть *централизованными*. Рекомендуемый порядок назначения IP-адресов дается в спецификации RFC 2050.

Централизованное распределение адресов

Когда дело касается сети, являющейся частью интернета, уникальность нумерации может быть обеспечена только усилиями специально созданных для этого центральных органов. Такие централизованно распределяемые адреса называются **глобальными/публичными адресами**.

В небольшой же *автономной* IP-сети условие уникальности номеров сетей и узлов может быть обеспечено «вручную» сетевым администратором. В этом случае в распоряжении администратора имеется все адресное пространство, ограниченное лишь разрядностью IP-адресов, так как совпадение IP-адресов в не связанных между собой сетях не вызовет никаких отрицательных последствий. Администратор может выбирать адреса произвольным образом, соблюдая лишь синтаксические правила и учитывая ограничения на особые адреса.

Но при таком подходе исключена возможность в будущем подсоединить данную сеть к интернету. Действительно, произвольно выбранные адреса данной сети могут совпасть с централизованно назначенными адресами других сетей, подключенных к интернету. Чтобы избежать коллизий, связанных с такого рода совпадениями, в стандартах интернета определено несколько диапазонов так называемых **частных адресов**, рекомендуемых для автономного использования:

- в классе A — сеть 10.0.0.0;
- в классе B — диапазон из 16 номеров сетей (172.16.0.0–172.31.0.0);
- в классе C — диапазон из 255 сетей (192.168.0.0–192.168.255.0).

Эти адреса, исключенные из множества централизованно распределяемых, составляют огромное адресное пространство, достаточное для нумерации узлов автономных сетей практически любых размеров. Заметим также, что частные адреса, как и при произвольном выборе адресов, в разных автономных сетях могут совпадать. В то же время с помощью специальных технологий можно избежать коллизий при подключении автономных сетей к интернету.

> В больших сетях, подобных Интернету, уникальность сетевых адресов гарантируется централизованной иерархически организованной системой их распределения. Номер сети может быть назначен только по рекомендации специального подразделения Интернета. Главным органом регистрации глобальных адресов в Интернете с 1998 года является неправительственная некоммерческая организация **ICANN** (Internet Corporation for Assigned Names and Numbers).

Эта организация координирует работу региональных отделов, деятельность которых охватывает большие географические площади: ARIN — Америка, RIPE (Европа), APNIC (Азия и Тихоокеанский регион). Региональные отделы выделяют блоки адресов сетей крупным поставщикам услуг, а те, в свою очередь, распределяют их между своими клиентами, среди которых могут быть и более мелкие поставщики.

Проблемой централизованного распределения адресов является их *дефицит*. Уже сравнительно давно очень трудно получить адрес класса B и невозможно — адрес класса A. Надо отметить, что дефицит обусловлен не только ростом количества узлов и сетей, но и тем, что имеющееся адресное пространство используется *нерационально*.

Для смягчения проблемы дефицита адресов разработчики стека TCP/IP предлагают разные подходы. Принципиальным решением является переход на новую версию протокола IP — *протокол IPv6*, в котором проблемы дефицита адресов не существует. Но и текущая версия протокола IP (IPv4) поддерживает технологии, направленные на более экономное расходование IP-адресов, такие, например, как NAT и CIDR.

Суть **технологии трансляции сетевых адресов NAT** (см. главу 30) состоит в следующем. Пусть предприятие получило от поставщика услуг небольшое число дефицитных публичных IP-адресов, которых явно недостаточно для нумерации всех узлов сети. В таком случае администраторы сети могут использовать для адресации практически неисчерпаемый пул частных адресов. Чтобы исключить коллизии, на маршрутизаторе, связывающем сеть предприятия с внешней сетью, устанавливается программное обеспечение, динамически отображающее (транслирующее) частные адреса на набор имеющихся в наличии публичных адресов, присвоенных внешнему интерфейсу маршрутизатора предприятия.

Технология бесклассовой маршрутизации CIDR

Технология бесклассовой междоменной маршрутизации (Classless Inter-Domain Routing, **CIDR**) основана[1] на использовании масок для более гибкого распределения адресов и более эффективной маршрутизации. Она допускает произвольное разделение IP-адреса на поля для нумерации сети и узлов. При такой системе адресации клиенту может быть выдан пул адресов, более точно соответствующий его запросу, чем это происходит при адресации, основанной на классах адресов.

Например, если *клиенту А* (рис. 13.5) требуется всего 13 адресов, то вместо выделения ему сети стандартного класса C (класса с наименьшим числом узлов — 256) ему может быть назначен пул адресов 193.20.30.0/28. Эта запись интерпретируется следующим образом: «сеть, не принадлежащая ни к какому стандартному классу, номер которой содержится в 28 старших двоичных разрядах IP-адреса 193.20.30.0, имеющая 4-битовое поле для нумерации 16 узлов». Все это вполне удовлетворяет требованиям клиента А. Очевидно, что такой вариант намного более экономичен, чем раздача сетей стандартных классов «целиком».

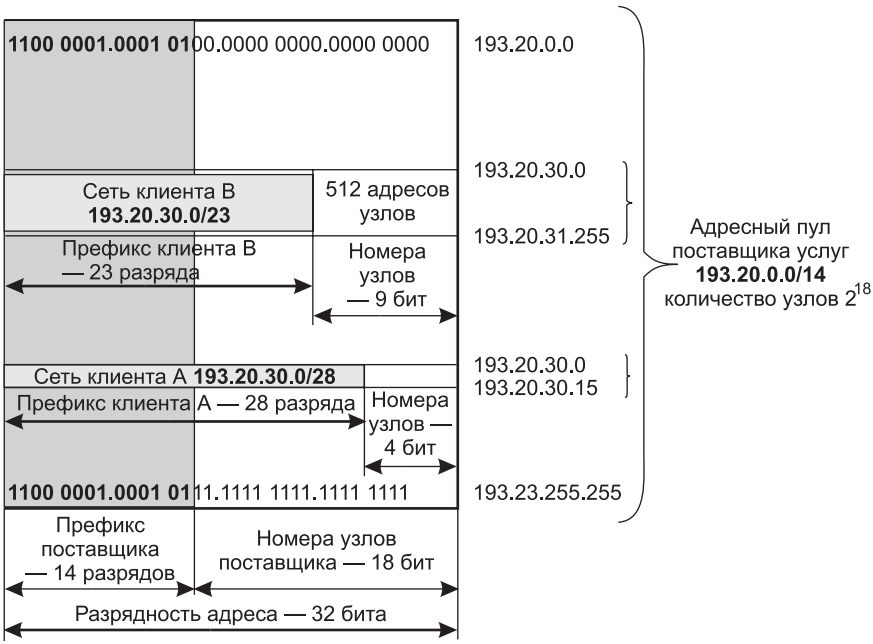


Рис. 13.5. Схема распределения адресного пространства в соответствии с CIDR

Определение пула адресов в виде пары *IP-адрес/маска* возможно только при выполнении нескольких условий. Прежде всего адресное пространство, из которого организация, распределяющая адреса, «нарезает» адресные пулы для заказчиков, должно быть *непре-*

[1] Технология CIDR описана в документах RFC 1517 — RFC 1520.

рывным. При таком условии все адреса имеют общий **префикс** — одинаковую последовательность цифр в старших разрядах адреса.

Пусть, например, как показано на рис. 13.5, провайдер располагает адресами в диапазоне 193.20.0.0–193.23.255.255 или в двоичной записи:

1100 0001.0001 0100.0000 0000.0000 0000–1100 0001.0001 0111.1111 1111.1111 1111.

Здесь префикс провайдера имеет длину 14 разрядов — 1100 0001.0001 01, что можно записать в виде193.20.0.0/14. Префикс обычно интерпретируется как *номер подсети.*

Назначение адресов в виде IP-адрес/маска корректно лишь в случае, если поле для адресации узлов, полученное применением маски к IP-адресу, содержит только нули. Например, определение пула адресов в виде 193.20.0.0/12 ошибочно, так как в поле номера сети (в 20 младших битах) содержится не нулевое значение 0100.0000 0000.0000 0000. В то же время префикс может оканчиваться нулями, например, определение пула 193.20.0.0/25, в котором префикс имеет значение 1100 0001.0001 0100.0000 0000.0, вполне корректно. Итак, для обобщенного представления пула адресов в виде *IP-адрес/n* справедливы следующие утверждения:

- значением префикса (номера сети) являются n старших двоичных разрядов IP-адреса;
- поле для адресации узлов состоит из $(32 - n)$ младших двоичных разрядов IP-адреса;
- первый по порядку номер узла должен состоять только из нулей;
- количество адресов в пуле равно $2^{(32-n)}$.

Благодаря CIDR провайдер получает возможность «нарезать» блоки из выделенного ему адресного пространства в соответствии с действительными требованиями каждого клиента (в главе 14 мы обсудим, как технология CIDR помогает не только экономно расходовать адреса, но и эффективнее осуществлять маршрутизацию).

Отображение IP-адресов на локальные адреса

Одной из главных задач, ставившихся при создании протокола IP, являлось обеспечение совместной согласованной работы сети, состоящей из подсетей, в общем случае использующих *разные сетевые технологии.* При перемещении IP-пакета по составной сети взаимодействие технологии TCP/IP с локальными технологиями подсетей происходит многократно. На каждом маршрутизаторе протокол IP определяет, какому следующему маршрутизатору в этой сети надо направить пакет. В результате решения этой задачи протоколу IP становится известен *IP-адрес* интерфейса следующего маршрутизатора (или конечного узла, если эта сеть является сетью назначения). Чтобы локальная технология сети смогла доставить пакет следующему маршрутизатору, необходимо:

- упаковать пакет в кадр соответствующего для данной сети формата (например, Ethernet);
- снабдить данный кадр *локальным адресом* следующего маршрутизатора.

Решением этих задач занимается уровень сетевых интерфейсов стека TCP/IP.

Итак, никакой функциональной зависимости между локальным адресом и его IP-адресом не существует, следовательно, единственный способ установления соответствия — ведение

таблиц. В результате конфигурирования сети каждый интерфейс «знает» свои IP-адрес и локальный адрес, что можно рассматривать как таблицу, состоящую из одной строки. Проблема состоит в том, как организовать обмен имеющейся информацией между узлами сети.

Протокол ARP

Для определения локального адреса по IP-адресу используется **протокол разрешения адресов** (Address Resolution Protocol, **ARP**). ARP реализуется различным образом в зависимости от того, работает ли он в локальной сети (Ethernet, Wi-Fi) с возможностью широковещания или же в глобальной сети (MPLS, ATM), которые, как правило, не поддерживают широковещательный доступ.

Рассмотрим работу протокола ARP в локальных сетях с *широковещанием*.

На рис. 13.6 показан фрагмент IP-сети, включающий две сети – Ethernet1 (из трех конечных узлов – *A*, *B* и *C*) и Ethernet2 (из двух конечных узлов – *D* и *E*). Сети подключены соответственно к интерфейсам 1 и 2 маршрутизатора. Каждый сетевой интерфейс имеет IP-адрес и MAC-адрес. Пусть в какой-то момент IP-модуль узла *C* направляет пакет узлу *D*. Протокол IP узла *C* определил IP-адрес интерфейса следующего маршрутизатора – это IP_1. Теперь, прежде чем упаковать пакет в кадр Ethernet и направить его маршрутизатору, необходимо определить соответствующий MAC-адрес. Для решения этой задачи *протокол IP обращается к протоколу ARP*.

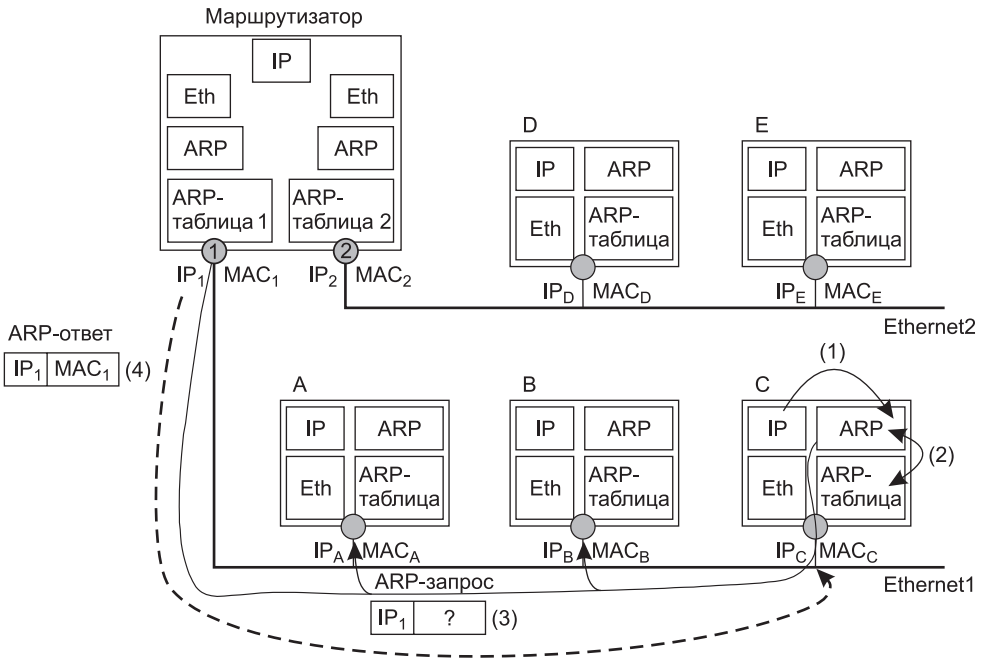


Рис. 13.6. Схема работы протокола ARP

Протокол ARP поддерживает на каждом интерфейсе сетевого адаптера или маршрутизатора отдельную ARP-таблицу, в которой в ходе функционирования сети накапливается информация о соответствии между IP-адресами и MAC-адресами других интерфейсов данной сети. Первоначально, при включении компьютера или маршрутизатора в сеть, все его ARP-таблицы пусты.

1. На первом шаге происходит передача от протокола IP протоколу ARP примерно такого сообщения: «Какой MAC-адрес имеет интерфейс с адресом IP_1?»
2. Работа протокола ARP начинается с просмотра собственной ARP-таблицы. Предположим, что среди содержащихся в ней записей отсутствует запрашиваемый IP-адрес.
3. В этом случае протокол ARP формирует **ARP-запрос**, вкладывает его в кадр протокола Ethernet и широковещательно рассылает. Заметим, что зона распространения ARP-запроса ограничивается сетью Ethernet1, так как на пути широковещательных кадров барьером стоит маршрутизатор.
4. Все интерфейсы сети Ethernet1 получают ARP-запрос и направляют его «своему» протоколу ARP. ARP сравнивает указанный в запросе адрес IP_1 с IP-адресом собственного интерфейса.
5. Протокол ARP, который констатировал совпадение (в данном случае это ARP интерфейса 1 маршрутизатора), формирует ARP-ответ, в котором маршрутизатор указывает локальный адрес MAC_1, соответствующий адресу IP_1 своего интерфейса, и отправляет его запрашивающему узлу (в данном примере узлу *C*).

На рис. 13.7 показан кадр Ethernet с вложенным в него ARP-сообщением. ARP-запросы и ARP-ответы имеют один и тот же формат. В табл. 13.3 в качестве примера приведены значения полей реального ARP-запроса, переданного по сети Ethernet[1].

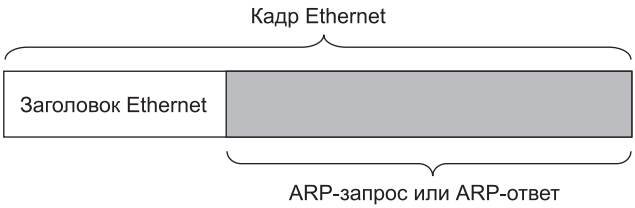


Рис. 13.7. Инкапсуляция ARP-сообщений в кадр Ethernet

В поле типа сети для сетей Ethernet указывается значение 1. Поле типа протокола позволяет использовать протокол ARP не только с протоколом IP, но и с другими сетевыми протоколами. Для IP значение этого поля равно 0x0800. Длина локального адреса для протокола Ethernet равна 6 байт, а длина IP-адреса — 4 байта. В поле операции для ARP-запросов указывается значение 1, для ARP-ответов — значение 2. Из запроса видно, что в сети Ethernet узел с IP-адресом 194.85.135.75 пытается определить, какой MAC-адрес имеет другой узел той же сети, сетевой адрес которого 194.85.135.65. *Поле искомого локального адреса заполнено нулями.*

[1] Символы 0x означают, что за ними следует число, записанное в шестнадцатеричном формате.

Таблица 13.3. Пример ARP-запроса

Поле	Значение
Тип сети	1 (0x1)
Тип протокола	2048 (0x800)
Длина локального адреса	6 (0x6)
Длина сетевого адреса	4 (0x4)
Операция	1 (0x1)
Локальный адрес отправителя	008048EB7E60
Сетевой адрес отправителя	194.85.135.75
Локальный (искомый) адрес получателя	000000000000
Сетевой адрес получателя	194.85.135.65

Ответ присылает узел, опознавший свой IP-адрес. Если в сети нет интерфейса с искомым IP-адресом, то ARP-ответа не будет. Протокол IP уничтожает IP-пакеты, направляемые по этому адресу. В табл. 13.4 показаны значения полей ARP-ответа, который мог бы поступить на приведенный в табл. 13.3 ARP-запрос.

Таблица 13.4. Пример ARP-ответа

Поле	Значение
Тип сети	1 (0x1)
Тип протокола	2048 (0x800)
Длина локального адреса	6 (0x6)
Длина сетевого адреса	4 (0x4)
Операция	2 (0x1)
Локальный адрес отправителя	00E0F77F1920
Сетевой адрес отправителя	194.85.135.65
Локальный (искомый) адрес получателя	008048EB7E60
Сетевой адрес получателя	194.85.135.75

В результате обмена ARP-сообщениями модуль IP, пославший запрос с интерфейса, имеющего адрес 194.85.135.75, определил, что IP-адресу 194.85.135.65 соответствует MAC-адрес 00E0F77F1920. Теперь операция упаковки IP-пакета в кадр Ethernet может быть успешно завершена.

Чтобы уменьшить число ARP-обращений, запись о найденном соответствии между IP-адресом и MAC-адресом *сохраняется* в ARP-таблице соответствующего интерфейса. Эта запись в ARP-таблице появляется автоматически, спустя несколько миллисекунд после того, как модуль ARP проанализирует ARP-ответ.

Протокол ARP, получив запрос, прежде всего просматривает свою ARP-таблицу и делает широковещательный запрос только в том случае, если в таблице нет записи о запрошенном адресе. ARP-таблица пополняется *не только за счет поступающих на данный интерфейс*

ARP-ответов, но и в результате извлечения полезной информации из широковещательных ARP-запросов. Действительно, в каждом запросе, как это видно из табл. 13.3 и 13.4, содержатся IP- и MAC-адреса отправителя. Все интерфейсы, получившие этот запрос, могут поместить информацию о соответствии локального и сетевого адресов отправителя в собственную ARP-таблицу. В частности, все узлы, получившие ARP-запрос (см. табл. 13.3), могут пополнить свою ARP-таблицу записью:

194.85.135.75 — 008048EB7E60

Таким образом, ARP-таблица, в которую в ходе работы сети были добавлены две упомянутые нами записи, может иметь вид (табл. 13.5):

Таблица 13.5. Пример ARP-таблицы

IP-адрес	MAC-адрес	Тип записи
194.85.135.65	00E0F77F1920	Динамический
194.85.135.75	008048EB7E60	Динамический
194.85.60.21	008048EB7567	Статический

В ARP-таблицах существуют два типа записей: динамические и статические. **Статические записи** создаются вручную с помощью утилиты **arp** и не имеют срока устаревания, точнее, они существуют до тех пор, пока компьютер или маршрутизатор остается включенным. **Динамические записи** должны периодически обновляться. Если запись не обновлялась в течение определенного времени (порядка нескольких минут), то она исключается из таблицы. Таким образом, в ARP-таблице содержатся записи не обо всех узлах сети, а только о тех, которые активно участвуют в сетевых операциях. Поскольку такой способ хранения информации называют кэшированием, ARP-таблицы иногда называют **ARP-кэшем**.

ПРИМЕЧАНИЕ

Некоторые реализации протоколов IP и ARP не ставят IP-пакеты в очередь на время ожидания ARP-ответов. Вместо этого IP-пакет просто уничтожается, а его восстановление возлагается на модуль TCP или прикладной процесс, работающий через протокол UDP. Такое восстановление выполняется за счет тайм-аутов и повторных передач. Успешность повторной передачи обеспечивается первой попыткой, которая вызывает заполнение ARP-таблицы.

Совсем другой способ разрешения адресов используется *в глобальных сетях*, в которых не поддерживается широковещательная рассылка. Здесь администратору сети чаще всего приходится вручную формировать и помещать на какой-либо сервер ARP-таблицу, в которой он задает, например, соответствие IP-адресов адресам X.25, имеющим для протокола IP смысл локальных адресов.

В то же время сегодня наметилась тенденция автоматизации работы протокола ARP и в глобальных сетях. Для этой цели среди всех маршрутизаторов, подключенных к какой-либо глобальной сети, выделяется специальный маршрутизатор — **ARP-сервер**, который ведет ARP-таблицу для всех остальных узлов и маршрутизаторов этой сети. При таком централизованном подходе единственное, что требуется сделать вручную, — это занести в память всех компьютеров и маршрутизаторов IP-адрес и локальный адрес ARP-сервера. При включении каждый узел и маршрутизатор регистрируют свой адрес в ARP-сервере.

Всякий раз, когда возникает необходимость определения по IP-адресу локального адреса, модуль ARP обращается к ARP-серверу с запросом и автоматически получает ответ.

В некоторых случаях возникает обратная задача — нахождение IP-адреса по известному локальному адресу. Тогда в действие вступает **реверсивный протокол разрешения адресов** (Reverse Address Resolution Protocol, **RARP**). Этот протокол используется, например, при старте бездисковых станций, не знающих в начальный момент времени своего IP-адреса, но знающих MAC-адрес своего сетевого адаптера.

Протокол Proxy-ARP

Протокол Proxy-ARP — это разновидность протокола ARP, позволяющая отображать IP-адреса на аппаратные адреса в сетях, поддерживающих широковещание, даже в тех случаях, когда искомый узел находится за пределами данного домена коллизий.

На рис. 13.8 показана сеть, один из конечных узлов которой (компьютер *D*) работает в **режиме удаленного узла**. В этом режиме узел обладает всеми возможностями компьютеров основной части сети Ethernet и, в частности, имеет IP-адрес (IP_D), относящийся к той же сети. Для всех конечных узлов сети Ethernet особенности подключения удаленного узла (наличие модемов, коммутируемая связь, протокол PPP) абсолютно прозрачны — они взаимодействуют с ним обычным образом. Чтобы такой режим взаимодействия стал

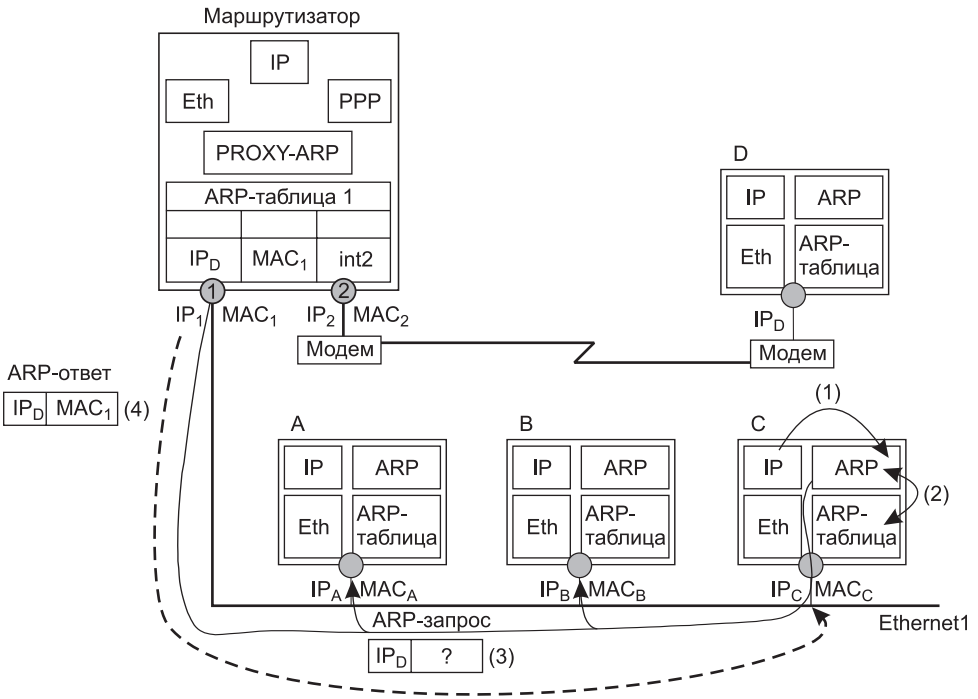


Рис. 13.8. Схема работы протокола Proxy-ARP

возможным, среди прочего необходим протокол Proxy-ARP. Поскольку удаленный узел подключен к сети по протоколу PPP, то он *не имеет MAC-адреса.*

Пусть приложение, работающее, например, на компьютере *C*, решает послать пакет компьютеру *D*. Ему известен IP-адрес узла назначения (IP_D), однако, как отмечено, для передачи пакета по сети Ethernet его необходимо упаковать в кадр Ethernet и снабдить MAC-адресом. Для определения MAC-адреса IP-протокол узла *C* обращается к протоколу ARP, который посылает широковещательное сообщение с ARP-запросом. Если бы в этой сети на маршрутизаторе не был установлен протокол Proxy-ARP, то на этот запрос не откликнулся бы ни один узел.

Однако протокол Proxy-ARP установлен на маршрутизаторе и работает следующим образом. При подключении к сети удаленного узла *D* в таблицу ARP-маршрутизатора заносится запись

$$IP_D - MAC_1 - int2$$

Эта запись означает, что:

- при поступлении ARP-запроса на маршрутизатор относительно адреса IPD в ARP-ответ будет помещен аппаратный адрес MAC_1, соответствующий аппаратному адресу интерфейса 1 маршрутизатора;
- узел, имеющий адрес IP_D, подключен к интерфейсу 2 маршрутизатора.

В ответ на посланный узлом *C* широковещательный ARP-запрос откликается маршрутизатор с установленным протоколом Proxy-ARP. Он посылает ложный ARP-ответ, в котором на место аппаратного адреса компьютера *D* помещает собственный адрес MAC_1. Узел *C*, не подозревая подвоха, посылает кадр с IP-пакетом по адресу MAC_1. Получив кадр, маршрутизатор с установленным протоколом Proxy-ARP понимает, что он направлен не ему (в пакете указан чужой IP-адрес) и поэтому надо искать адресата в ARP-таблице. Из таблицы видно, что кадр надо направить узлу, подключенному ко второму интерфейсу.

Мы рассмотрели простейшую схему применения протокола Proxy-ARP, которая, тем не менее, достаточно полно отражает логику его работы.

Доменная служба имен DNS

Доменная служба имен (Domain Name Service, **DNS**) отображает символьные имена узлов сети на их IP-адреса (как IPv4, так и IPv6). Доменная служба имен является важной частью Интернета, но она может работать и в любой автономной IP-сети.

Пространство DNS-имен

В операционных системах, первоначально предназначенных для локальных сетей (Novell NetWare, Microsoft Windows или IBM OS/2), пользователи всегда работали с символьными именами компьютеров. Так как локальные сети состояли из небольшого числа компьютеров, применялись так называемые **плоские имена**, состоящие из последовательности символов, не разделенных на части. Примерами таких имен являются NW1_1, mail2, MOSCOW_SALES_2. Для установления соответствия между символьными именами и MAC-адресами в этих операционных системах применялся механизм широковещательных за-

просов, подобный механизму запросов протокола ARP. Так, широковещательный способ разрешения имен реализован в протоколе NetBIOS, на котором были построены многие локальные ОС. Так называемые NetBIOS-имена стали на долгие годы одним из основных типов плоских имен в локальных сетях.

Для стека TCP/IP, рассчитанного в общем случае на работу в больших территориально распределенных сетях, подобный подход оказывается неэффективным. В стеке TCP/IP применяется доменная система имен, которая имеет иерархическую древовидную структуру, допускающую наличие в имени произвольного количества составных частей (рис. 13.9).

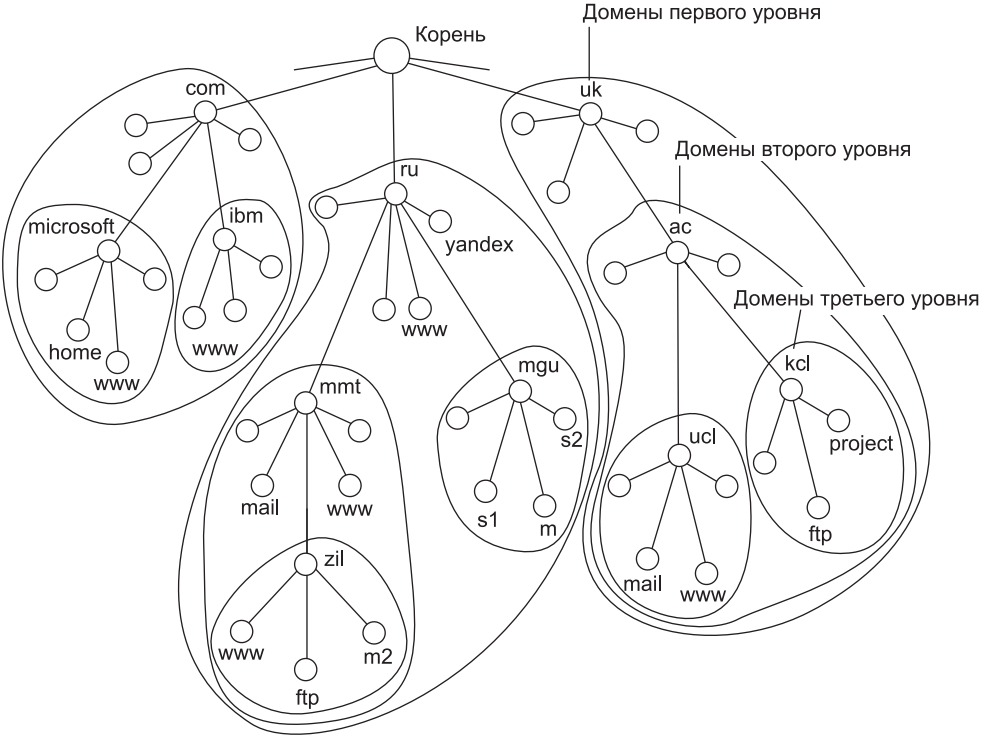


Рис. 13.9. Пространство доменных имен

Иерархия доменных имен аналогична иерархии имен файлов, принятой во многих популярных файловых системах. Дерево имен начинается с **корня**, обозначаемого здесь точкой. Затем следует старшая символьная часть имени, вторая по старшинству символьная часть имени и т. д. Младшая часть имени соответствует конечному узлу сети. В отличие от имен файлов, при записи которых сначала указывается самая старшая составляющая, затем составляющая более низкого уровня и т. д., запись доменного имени начинается с самой младшей составляющей, а заканчивается самой старшей. Составные части доменного имени отделяются друг от друга точкой. Например, в имени home.microsoft.com составляющая home является именем одного из компьютеров в домене microsoft.com.

Разделение имени на части позволяет *разделить административную ответственность* за назначение уникальных имен между различными людьми или организациями в пределах своего уровня иерархии. Так, для примера, приведенного на рис. 13.9, некоторая организация

может нести ответственность за то, чтобы все имена с окончанием «ru» имели уникальную следующую вниз по иерархии часть. То есть все имена типа www.ru, mail.mmt.ru или m2.zil.mmt.ru отличаются второй по старшинству частью. Разделение административной ответственности позволяет решить проблему образования уникальных имен без взаимных консультаций между организациями, отвечающими за имена одного уровня иерархии. Очевидно, что должна существовать одна организация, отвечающая за назначение имен верхнего уровня иерархии.

Совокупность имен, у которых несколько старших составных частей совпадают, образуют **домен имен** (name domain). Например, имена www.zil.mmt.ru, ftp.zil.mmt.ru, yandex.ru и s1.mgu.ru входят в домен ru, так как все они имеют одну общую старшую часть — имя ru. Другим примером является домен mgu.ru. Из представленных на рис. 13.9 имен в него входят имена s1.mgu.ru, s2.mgu.ru и m.mgu.ru. Этот домен образуют имена, у которых две старшие части равны mgu.ru. Администратор домена mgu.ru несет ответственность за уникальность имен следующего уровня, входящих в домен, то есть имен s1, s2 и m. Образованные домены s1.mgu.ru, s2.mgu.ru и m.mgu.ru являются **поддоменами** домена mgu. u, так как имеют общую старшую часть имени. Часто поддомены для краткости называют только младшей частью имени, то есть в нашем случае поддоменами являются s1, s2 и m.

Как и в файловой системе, в доменной системе имен различают краткие, относительные и полные доменные имена. **Краткое доменное имя** — это имя конечного узла сети: хоста или порта маршрутизатора. Краткое имя — это лист дерева имен. **Относительное доменное имя** — это составное имя, начинающееся с некоторого уровня иерархии, но не самого верхнего. Например, www.zil — это относительное имя. **Полное доменное имя** включает составляющие всех уровней иерархии, начиная от краткого имени и кончая корневой точкой: www.zil.mmt.ru. Если в каждом домене и поддомене обеспечивается уникальность имен следующего уровня иерархии, то и вся система имен будет состоять из уникальных имен.

ВНИМАНИЕ

Компьютеры, имена которых относятся к одному и тому же домену, могут иметь абсолютно независимые друг от друга IP-адреса, принадлежащие разным сетям и подсетям. Например, в домен mgu.ru могут входить хосты с адресами 132.13.34.15, 201.22.100.33 и 14.0.0.6.

Корневой домен имен управляется центральными органами интернета, в частности, такой организацией, как ICANN.

Домены верхнего уровня назначаются для каждой страны, а также для различных типов организаций. Для обозначения стран используются трехбуквенные и двухбуквенные аббревиатуры, например, ru (Россия), uk (Великобритания), fi (Финляндия), us (Соединенные Штаты), а для различных типов организаций, например, следующие обозначения:

- com — коммерческие организации (например, microsoft.com);
- edu — образовательные организации (например, mit.edu);
- gov — правительственные организации (например, nsf.gov);
- org — некоммерческие организации (например, fidonet.org);
- net — сетевые организации (например, nsf.net).

Каждый домен администрирует отдельная организация, которая обычно разбивает свой домен на поддомены и передает функции администрирования этих поддоменов другим организациям.

Сервер, клиент и протокол DNS

Широковещательный способ установления соответствия между символьными именами и локальными адресами, подобный реализованному в протоколе ARP, хорошо работает только в небольшой локальной сети, не разделенной на подсети. В крупных сетях, где возможность всеобщей широковещательной рассылки не поддерживается, нужен другой способ разрешения символьных имен. Альтернативой широковещательной рассылке является применение *централизованной службы*, поддерживающей соответствие между символьными именами и IP-адресами всех компьютеров сети. На раннем этапе развития интернета на каждом хосте вручную создавался текстовый **файл отображений** с известным именем hosts.txt. Этот файл состоял из некоторого количества строк, каждая из которых содержала одну пару «доменное имя — IP-адрес», например:

```
rhino.acme.com — 102.54.94.97
```

По мере роста интернета файлы hosts.txt также увеличивались в объеме, и создание *масштабируемого* решения для отображения имен стало необходимостью. Таким решением стала централизованная справочная служба — **система доменных имен** (Domain Name System, **DNS**), основанная на распределенной базе отображений «доменное имя — IP-адрес».

Как и всякая сетевая служба, система DNS состоит из серверов и клиентов. **DNS-серверы** поддерживают распределенную базу отображений, а **DNS-клиенты** обращаются к серверам с запросами об отображении доменного имени на IP-адрес (эту процедуру называют также «*разрешением*» доменного имени).

DNS-клиентом является практически каждый узел интернета, будь то клиентский компьютер, сервер приложений или маршрутизатор. Программа, реализующая функции клиента DNS, называется **резольвером** и обычно входит в состав ОС. Приложение (например, веб-браузер), когда у него возникает необходимость в отображении доменного имени (например, имени сайта), не делает запрос напрямую к службе DNS, а обращается к резольверу своей ОС, который затем взаимодействует с DNS-сервером. Резольверу должен быть известен IP-адрес, по крайней мере одного DNS-сервера. Этот адрес конфигурируется вручную или же получается по протоколу DHCP, о котором мы будем говорить в следующем разделе.

Подавляющее большинство DNS-серверов использует программное обеспечение **BIND** (Berkeley Internet Name Domain), разработанное в Калифорнийском университете Беркли.

Сервер и клиент службы DNS взаимодействуют по **протоколу DNS**. Сообщения этого протокола чаще всего передаются в *дейтаграммах UDP* с портом сервера 53, но в некоторых случаях, требующих повышенной надежности, служба DNS обращается к услугам TCP.

Иерархическая организация службы DNS

Иерархию образуют DNS-серверы. На самой вершине иерархии располагаются **корневые серверы**. Они хранят текстовые файлы имен и IP-адресов DNS-серверов следующего уровня, называемого *верхним* (top level DNS, tlDNS или TLD). Серверы верхнего уровня хранят данные об именах и адресах имен, входящих в домены верхнего уровня (com, ru или fm), а также об именах серверов DNS, которые обслуживают домены второго уровня иерархии (cisco.com, yandex.ru и т. п.).

Сервер DNS отвечает на запросы клиентов на основе информации, содержащейся в текстовых файлах отображений имен, хранящихся на данном сервере. В принципе, сервер DNS мог бы хранить данные всех отображений, входящих в некоторый домен со всеми его поддоменами. При таком подходе сервер верхнего уровня, отвечающий, например, за домен com, хранил бы в своих файлах записи всех имен, кончающихся на com: ibm.com, www.ibm.com, www2.ibm.com, cisco.com, www.cisco.com и т. д. Понятно, что такой подход не масштабируем и не может работать в глобальной сети.

Поэтому пространство доменных имен «разрезают» между серверами DNS, обычно так, чтобы сервер DNS хранил записи только в пределах одного уровня, а для имен своих поддоменов хранил только ссылки на серверы DNS, которые отвечают за эти поддомены. Например, DNS-сервер верхнего уровня, отвечающий за домен com, хранит только записи листьев своего домена, например имени www.com, а также имен серверов DNS, которые обслуживают поддомены домена com, например DNS-сервера поддомена cisco.com.

Часть пространства доменных имен, для которых некоторый сервер DNS имеет информацию об их отображениях на основе соответствующего текстового файла, называется **зоной DNS** данного сервера, а сам текстовый файл — **файлом зоны**.

Когда сервер DNS дает ответ о записи, входящей в зону, за которую он отвечает, такой ответ называется **полномочным ответом** DNS (authoritative, что можно также перевести как официальный, авторитетный или аутентичный). Как мы увидим далее, сервер DNS может также давать **неполномочный ответ**, если запрос относится не к его зоне, но он знает ответ на него за счет кэширования ответов других серверов.

Файл зоны состоит из текстовых записей нескольких типов:

- *запись типа A* — отображает имя в IPv4-адрес;
- *запись типа AAAA* — отображает имя в IPv6-адрес;
- *запись типа NS* — определяет имя DNS-сервера для некоторого домена;
- *запись типа MS* — определяет имя почтового сервера для некоторого домена,

а также некоторых других.

Протокол DNS позволяет клиенту делать запросы относительно некоторого доменного имени, задавая тип записи или же запрашивая все типы, относящиеся к данному имени. В системе DNS поддерживает не только отображение типа «один к одному», но и отображения «многие к одному» и «один ко многим». То есть хост может иметь, кроме одного основного доменного имени, несколько так называемых *псевдонимов* (alias). Такое свойство оказывается полезным, например, когда владелец коммерческого сайта не хочет терять клиентов из-за того, что они могут ошибиться при наборе имени, поэтому он поддерживает в файле зоны DNS некоторый набор похожих имен, которые все отображаются на один IP-адрес.

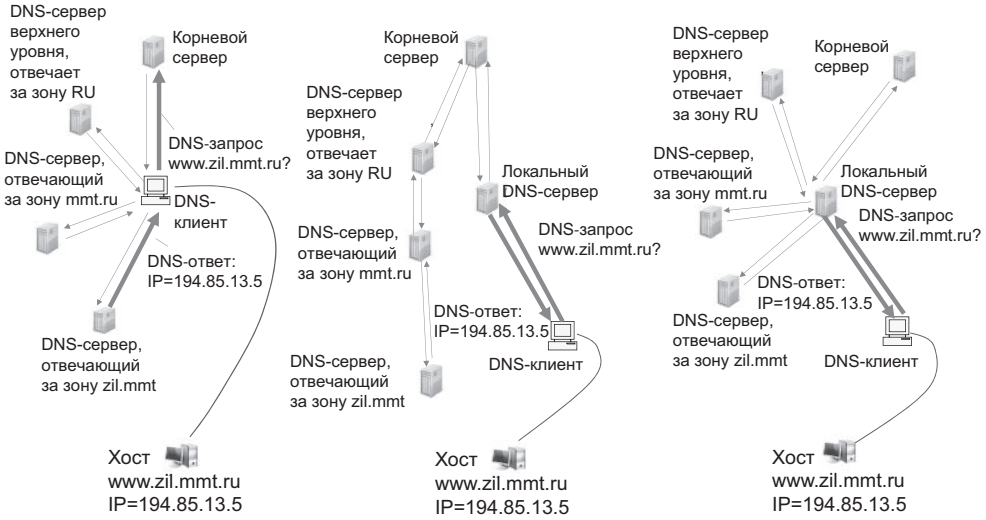
Отображение «один ко многим» используется для *распараллеливания нагрузки* на веб-серверы, которые испытывают перегрузку от наплыва запросов. Для этого перегруженный сервер заменяют несколькими функционально подобными серверами. В файле зоны DNS создается запись, в которой доменному имени данного веб-сервера ставится в соответствие набор IP-адресов дублирующих серверов. В ответе на каждый запрос DNS-сервер посылает весь набор IP-адресов, предварительно сдвинув его на одну позицию, поэтому нагрузка распределяется между всеми дублирующими серверами. Для обеспечения надежности и высокой производительности для каждой зоны существует один **первичный** (primary) **сервер DNS** и несколько **вторичных** (secondary) **серверов DNS**. На первичном

сервере находится исходный файл зоны — *мастер-копия файла зоны*, которая редактируется администратором сервера. Вторичные серверы периодически копируют файл зоны с первичного сервера. Для этого может использоваться протокол DNS, в котором имеется соответствующий тип запроса, или же администратор может использовать любой протокол копирования файлов, например ftp или scp. Если файл зоны передается по протоколу DNS, то для повышения надежности используется *протокол TCP* (порт 53).

Итеративная и рекурсивная процедуры разрешения имени

Существуют две основные схемы разрешения DNS-имен. В первом варианте, называемом **итеративной процедурой**, работу по поиску IP-адреса координирует *DNS-клиент*: он итеративно выполняет последовательность запросов к разным серверам имен. Рассмотрим эту процедуру на примере (рис. 13.10, *а*):

1. DNS-клиент обращается к корневому DNS-серверу с указанием полного доменного имени www.zil.mmt.ru хоста, для которого он хочет найти IP-адрес.
2. Корневой DNS-сервер отвечает клиенту, указывая адреса DNS-серверов верхнего уровня, обслуживающих домен, заданный в старшей части запрошенного имени, в данном случае — домен ru.
3. DNS-клиент делает следующий запрос к одному из предложенных ему DNS-серверов верхнего уровня, который отсылает его к DNS-серверу нужного поддомена (в примере это сервер, отвечающий за зону mmt.ru), и так далее, пока не будет найден DNS-сервер, в котором хранится отображение запрошенного имени на IP-адрес. Этот сервер дает окончательный ответ клиенту, который теперь может установить связь с хостом по IP-адресу 194.85.13.5.



а) Итеративная процедура *б*) Рекурсивная процедура *в*) Смешанная процедура

Рис. 13.10. Процедуры разрешения имен

Во втором варианте (рис. 13.10, *б*) выполняется **рекурсивная процедура**. Здесь DNS-клиент перепоручает всю работу по разрешению имени *цепочке DNS-серверов*:

1. DNS-клиент отправляет запрос к **локальному DNS-серверу**, то есть серверу, обслуживающему поддомен, которому принадлежит имя клиента.
2. Если локальный DNS-сервер знает ответ, то он сразу же возвращает его клиенту. Это может быть ***полномочный*** ответ (запрошенное имя входит в тот же поддомен, что и имя клиента) или ***неполномочный*** ответ (сервер уже узнавал данное соответствие для другого клиента и сохранил его в своем кэше).
3. Если локальный DNS-сервер не знает ответа, то он обращается к корневому серверу, который переправляет запрос к DNS-серверу верхнего уровня (отвечающему за зону RU), который в свою очередь запрашивает нижележащий сервер (зона mmt), и так далее, пока запрос не дойдет до полномочного сервера, имеющего в своем файле зоны запись о запрошенном имени.
4. DNS-ответ полномочного сервера проходит тот же путь по цепочке DNS-серверов в обратном направлении, пока не достигнет DNS-клиента, породившего данный запрос.

В третьем варианте (рис. 13.10, *в*) реализуется смешанная процедура, включающая рекурсивную и итеративную фазы:

1. Начальная часть процедуры, когда DNS-клиент передает запрос локальному DNS-серверу и поручает ему действовать от его имени, является *рекурсивной*.
2. Затем, если локальный DNS-сервер не знает ответа, то он последовательно выполняет ***итеративные*** запросы к иерархии серверов точно так же, как это делал DNS-клиент в первом варианте. Получив ответ, локальный DNS-сервер передает его клиенту.

DNS-серверы стараются не поддерживать рекурсивный режим ответов, так как это перегружает их; корневые серверы и серверы верхнего уровня всегда дают нерекурсивные ответы, отсылая серверы нижних уровней к серверам промежуточных уровней. Получая окончательный ответ от сервера вышестоящего уровня, рекурсивный сервер кэширует его для того, чтобы при поступлении аналогичного запроса дать быстрый неполномочный ответ. Чтобы служба DNS могла оперативно отрабатывать изменения, происходящие в сети, ответы кэшируются на относительно короткое время — обычно от нескольких часов до нескольких дней (срок задается администратором полномочного сервера). DNS-сервер может быть ***открытым*** (в этом случае он отвечает любому клиенту) или ***закрытым*** (в этом случае он отвечает либо только клиентам своего предприятия (в случае корпоративного сервера), либо только своим подписчикам услуг доступа в интернет (в случае провайдера)).

Корневые серверы

Корневые серверы — наиболее уязвимое звено[1] службы DNS, так как разрешение всех запросов, ответы на которые не находятся в кэше или файле зоны какого-либо DNS-сервера нижнего уровня, начинаются с обращения к одному из корневых серверов. Разработчики системы DNS (в начале 80-х годов) понимали это, поэтому изначально было решено обеспечить высокую степень резервирования: было установлено 13 корневых серверов

[1] Об атаках и методах защиты DNS-серверов читайте в главе 31.

с именами a.root-servers.net, b.root-servers.net, c.root-servers.net,... m.root-servers.net и тринадцатью IP-адресами.

С тех пор организация корневых DNS-серверов изменилась. Вместо 13 серверов интернет обслуживает более 300 — в августе 2013 года их было 376 (см. их географическое распределение на рис. 13.11). Это значительно повысило отказоустойчивость и производительность службы DNS. Все корневые серверы по-прежнему разделяют те же 13 имен (от a.root-servers.org до m.root-servers.org) и 13 IP-адресов. Но теперь каждому имени и адресу соответствует *кластер* серверов. Например, имени f.root-servers.net соответствует 56 серверов, а имени l.root-servers.net — 146. Корневые серверы распределены географически, а каждый кластер, соответствующий одному имени, администрируется отдельной организацией.

Обратная зона

Служба DNS предназначена не только для нахождения IP-адреса по имени хоста, но и для решения *обратной задачи* — нахождения DNS-имени по известному IP-адресу.

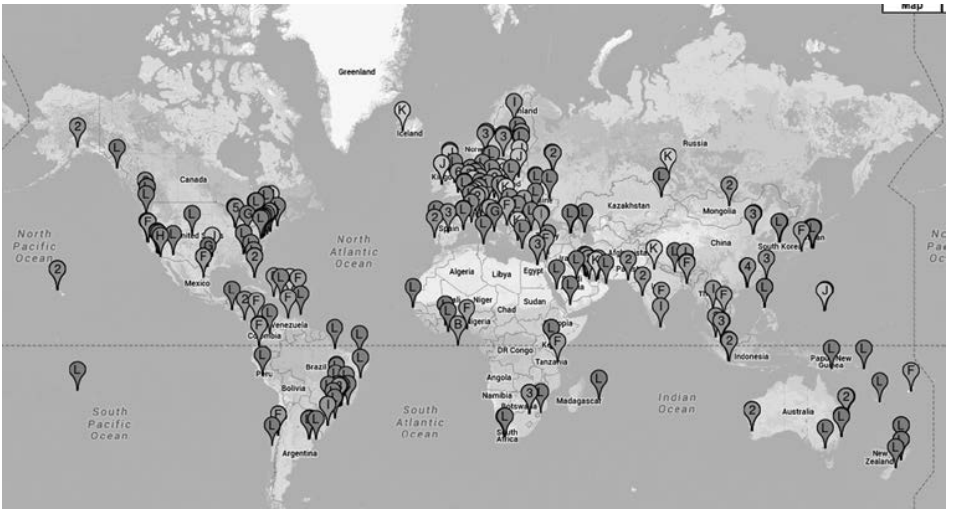


Рис. 13.11. Географическое распределение корневых серверов DNS (источник: root-servers.org)

Многие программы и утилиты, пользующиеся службой DNS, пытаются найти имя узла по его адресу в том случае, когда пользователем задан только адрес (или этот адрес программа узнала из пришедшего пакета). Обратная запись не всегда существует даже для тех адресов, для которых есть прямые записи. Ее могут просто забыть создать или же ее создание требует дополнительной оплаты. Обратная задача решается в интернете путем организации так называемых обратных зон.

Обратная зона — это система таблиц, которая хранит соответствие между IP-адресами и DNS-имена хостов некоторой сети. Для организации распределенной службы и использования для поиска имен того же программного обеспечения, что и для поиска адресов, применяется оригинальный подход, связанный с представлением IP-адреса в виде DNS-имени.

Первый этап преобразования заключается в том, что составляющие IP-адреса интерпретируются как составляющие DNS-имени. Например, адрес 192.31.106.0 рассматривается как состоящий из старшей части, соответствующей домену 192, затем идет домен 31, в который входит домен 106.

Далее, учитывая, что при записи IP-адреса старшая часть является самой *левой* частью адреса, а при записи DNS-имени — самой *правой*, то составляющие в преобразованном адресе указываются в обратном порядке, то есть для данного примера — 106.31.192. Для хранения отображений всех адресов, начинающихся, например, с числа 192, заводится зона 192 со своими серверами имен. Для записей о серверах, поддерживающих старшие в иерархии обратные зоны, создана специальная зона in-addr.arpa, поэтому полная запись для использованного в примере адреса выглядит так:

```
106.31.192.in-addr.arpa
```

Серверы для обратных зон используют файлы баз данных, не зависящие от файлов основных зон, в которых имеются записи о прямом соответствии тех же имен и адресов. Такая организация данных может приводить к несогласованности, так как одно и то же соответствие вводится в файлы дважды.

Протокол DHCP

Для нормальной работы сети каждому сетевому интерфейсу компьютера и маршрутизатора должен быть назначен IP-адрес.

Процедура присвоения адресов происходит в ходе **конфигурирования** компьютеров и маршрутизаторов. Назначение IP-адресов может происходить вручную в результате выполнения процедуры конфигурирования интерфейса, для компьютера сводящейся, например, к заполнению системы экранных форм. При этом администратор должен помнить, какие адреса из имеющегося множества он уже использовал для других интерфейсов, а какие еще свободны. При конфигурировании помимо IP-адресов сетевых интерфейсов (и соответствующих масок) устройству сообщается ряд других **конфигурационных параметров**. При конфигурировании администратор должен назначить клиенту не только IP-адрес, но и другие параметры стека TCP/IP, необходимые для его эффективной работы, например маску и IP-адрес маршрутизатора, предлагаемые по умолчанию, IP-адрес DNS-сервера, доменное имя компьютера и т. п. Даже при не очень большом размере сети эта работа представляет для администратора утомительную процедуру.

Протокол динамического конфигурирования хостов (Dynamic Host Configuration Protocol, **DHCP**) автоматизирует процесс конфигурирования сетевых интерфейсов, гарантируя от дублирования адресов за счет централизованного управления их распределением.

Режимы DHCP

Протокол DHCP работает в соответствии с моделью *клиент — сервер*. Во время старта ОС компьютер, являющийся DHCP-клиентом, посылает в сеть широковещательный запрос на получение IP-адреса. DHCP-сервер откликается и посылает сообщение-ответ, содержащее

IP-адрес и ряд других конфигурационных параметров. При этом DHCP-сервер может работать в разных режимах, включая:

- ручное назначение статических адресов;
- автоматическое назначение статических адресов;
- автоматическое распределение динамических адресов.

Во всех режимах работы администратор при конфигурировании DHCP-сервера сообщает ему один или несколько диапазонов IP-адресов, причем все эти адреса относятся к *одной сети*, то есть имеют одно и то же значение в поле номера сети.

В **ручном** режиме администратор, помимо пула доступных адресов, снабжает DHCP-сервер информацией о жестком соответствии IP-адресов физическим адресам или другим идентификаторам клиентских узлов. DHCP-сервер, пользуясь этой информацией, *всегда* выдаст определенному DHCP-клиенту *один и тот же* назначенный ему администратором IP-адрес (а также набор других конфигурационных параметров[1]).

В режиме **автоматического** назначения статических адресов DHCP-сервер самостоятельно, без вмешательства администратора произвольным образом выбирает клиенту IP-адрес из пула наличных IP-адресов. Адрес дается клиенту из пула в *постоянное* пользование, то есть между идентифицирующей информацией клиента и его IP-адресом по-прежнему, как и при ручном назначении, существует постоянное соответствие. Оно устанавливается в момент первого назначения DHCP-сервером IP-адреса клиенту. При последующих запросах сервер возвращает клиенту тот же самый IP-адрес.

При **динамическом** распределении адресов DHCP-сервер выдает адрес клиенту на *ограниченное время*, называемое **сроком аренды**. Когда компьютер, являющийся DHCP-клиентом, удаляется из подсети, назначенный ему IP-адрес автоматически освобождается. Когда компьютер подключается к другой подсети, то ему автоматически назначается новый адрес. Ни пользователь, ни сетевой администратор не вмешиваются в этот процесс. Это дает возможность впоследствии повторно использовать этот IP-адрес для назначения другому компьютеру. Таким образом, помимо основного преимущества DHCP — автоматизации рутинной работы администратора по конфигурированию стека TCP/IP на каждом компьютере, режим динамического распределения адресов в принципе позволяет строить IP-сеть, количество узлов в которой превышает количество имеющихся в распоряжении администратора IP-адресов.

Покажем преимущества, которые дает динамическое распределение пула адресов на примере. Пусть сотрудники некоторой организации значительную часть рабочего времени проводят вне офиса — дома или в командировках. Каждый из них имеет портативный компьютер, который во время пребывания в офисе подключается к корпоративной IP-сети. Возникает вопрос: сколько IP-адресов необходимо этой организации?

Первый ответ — столько, *скольким сотрудникам необходим доступ в сеть*. Если их 500 человек, то каждому из них должен быть назначен IP-адрес и выделено рабочее место. То есть администрация должна получить у поставщика услуг адреса двух сетей класса C и оборудовать соответствующим образом помещение. Однако вспомним, что сотрудники в этой организации редко появляются в офисе, и, значит, большая часть ресурсов при таком решении будет простаивать.

---

[1] Далее для краткости это уточнение будет опускаться.

Второй ответ — столько, *сколько сотрудников обычно присутствует в офисе* (с некоторым запасом). Если обычно в офисе работает не более 50 сотрудников, то достаточно получить у поставщика услуг пул из 64 адресов и установить в рабочем помещении сеть с 64 коннекторами для подключения компьютеров. Но возникает другая проблема — кто и как будет конфигурировать компьютеры, состав которых постоянно меняется?

Существует два пути. Во-первых, администратор (или сам мобильный пользователь) может конфигурировать компьютер вручную каждый раз, когда возникает необходимость подключения к офисной сети. Такой подход требует от администратора (или пользователей) большого объема рутинной работы — словом, это плохое решение. Второй вариант — автоматическое динамическое назначение DHCP-адресов. Действительно, администратору достаточно один раз при настройке DHCP-сервера указать диапазон из 64 адресов, а каждый новый мобильный пользователь будет просто физически подключать в сеть свой компьютер, на котором запускается DHCP-клиент. Запросив конфигурационные параметры, он автоматически получит их от DHCP-сервера. Таким образом, для работы 500 мобильных сотрудников достаточно иметь в офисной сети 64 IP-адреса и 64 рабочих места.

Динамическое назначение адресов

Администратор управляет процессом конфигурирования сети, определяя два основных конфигурационных параметра DHCP-сервера: *пул адресов, доступных распределению*, и *срок аренды*. Срок аренды диктует, как долго компьютер может использовать назначенный IP-адрес, перед тем как снова запросить его у DHCP-сервера. Срок аренды зависит от режима работы пользователей сети. Если это небольшая сеть учебного заведения, куда со своими компьютерами приходят многочисленные студенты для выполнения лабораторных работ, то срок аренды может быть равен длительности лабораторной работы. Если же это корпоративная сеть, в которой сотрудники предприятия работают на регулярной основе, то срок аренды может быть достаточно длительным — несколько дней или даже недель.

DHCP-сервер должен находиться *в одной подсети с клиентами*, учитывая, что клиенты посылают ему широковещательные запросы (рис. 13.12). Для снижения риска выхода сети из строя из-за отказа DHCP-сервера в сети иногда ставят резервный DHCP-сервер (такой вариант соответствует сети 1).

Иногда наблюдается и обратная картина: в сети нет ни одного DHCP-сервера. В этом случае его подменяет связной **DHCP-агент** — программное обеспечение, играющее роль посредника между DHCP-клиентами и DHCP-серверами (пример такого варианта — сеть 2). Связной агент переправляет запросы клиентов из сети 2 DHCP-серверу сети 3. Таким образом, один DHCP-сервер может обслуживать DHCP-клиентов нескольких разных сетей. Вот как выглядит упрощенная схема обмена сообщениями между клиентскими и серверными частями DHCP:

1. Когда компьютер включают, установленный на нем DHCP-клиент посылает ограниченное широковещательное сообщение DHCP-поиска (IP-пакет с адресом назначения, состоящим из одних единиц, который должен быть доставлен всем узлам данной IP-сети).
2. Находящиеся в сети DHCP-серверы получают это сообщение. Если в сети DHCP-серверы отсутствуют, то сообщение DHCP-поиска получает связной DHCP-агент. Он пересылает это сообщение в другую, возможно, значительно отстоящую от него сеть DHCP-серверу, IP-адрес которого ему заранее известен.

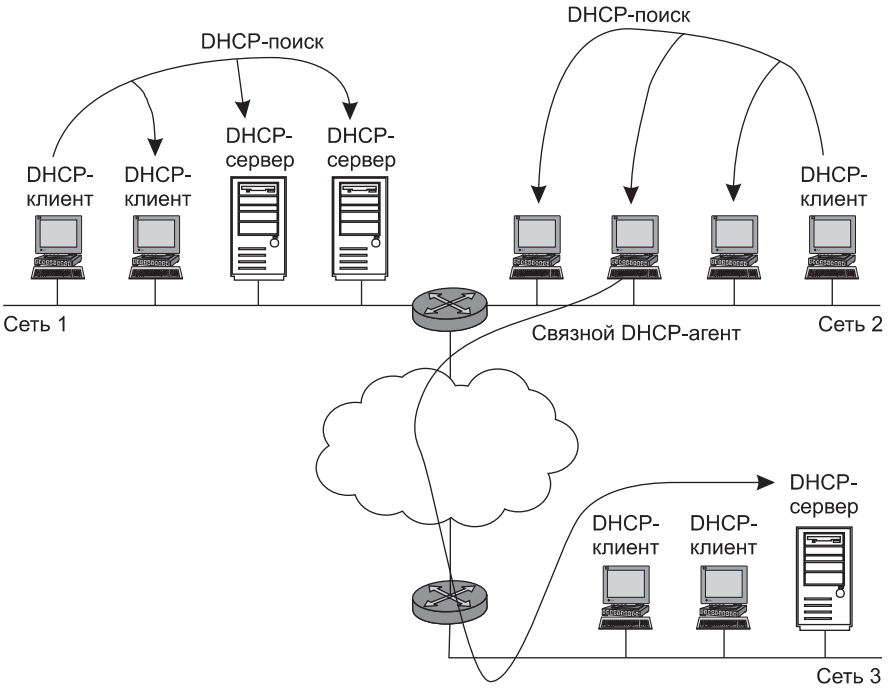


Рис. 13.12. Схемы взаимного расположения DHCP-серверов и DHCP-клиентов

3. Все DHCP-серверы, получившие сообщение DHCP-поиска, посылают DHCP-клиенту, обратившемуся с запросом, свои DHCP-предложения. Каждое предложение содержит IP-адрес и другую конфигурационную информацию. (DHCP-сервер, находящийся в другой сети, посылает ответ через агента.)
4. DHCP-клиент собирает конфигурационные DHCP-предложения от всех DHCP-серверов. Как правило, он выбирает первое из поступивших предложений и отправляет в сеть широковещательный DHCP-запрос. В этом запросе содержатся идентификационная информация о DHCP-сервере, предложение которого принято, а также значения принятых конфигурационных параметров.
5. Все DHCP-серверы получают DHCP-запрос и только один выбранный DHCP-сервер посылает положительную DHCP-квитанцию (подтверждение IP-адреса и параметров аренды), а остальные серверы аннулируют свои предложения, в частности, возвращают в свои пулы предложенные адреса.
6. DHCP-клиент получает положительную DHCP-квитанцию и переходит в рабочее состояние.

Время от времени компьютер пытается обновить параметры аренды у DHCP-сервера. Первую попытку он делает задолго до истечения срока аренды, обращаясь к тому серверу, от которого он получил текущие параметры. Если ответа нет или ответ отрицательный, то он через некоторое время снова посылает запрос. Так повторяется несколько раз, и если все попытки получить параметры у того же сервера оказываются безуспешными, то клиент обращается к другому серверу. Если и другой сервер отвечает отказом, то клиент теряет свои

конфигурационные параметры и переходит в режим автономной работы. Также DHCP-клиент может по своей инициативе досрочно отказаться от выделенных ему параметров.

В сети, где адреса назначаются динамически, нельзя быть уверенными в адресе, который в данный момент имеет тот или иной узел. Такое непостоянство IP-адресов влечет за собой некоторые проблемы.

Во-первых, *возникают сложности при преобразовании символьного доменного имени в IP-адрес.* Действительно, представьте себе функционирование системы DNS, которая должна поддерживать таблицы соответствия символьных имен IP-адресам в условиях, когда последние меняются каждые два часа! Учитывая это обстоятельство, для серверов, к которым пользователи часто обращаются по символьному имени, назначают статические IP-адреса, оставляя динамические только для клиентских компьютеров. Но в некоторых сетях количество серверов настолько велико, что их ручное конфигурирование становится слишком обременительным. Это привело к разработке усовершенствованной версии DNS (так называемой динамической системы DNS), в основе которой лежит согласование информационной адресной базы в службах DHCP и DNS.

Во-вторых, *трудно осуществлять удаленное управление и автоматический мониторинг интерфейса* (например, сбор статистики), если в качестве его идентификатора выступает динамически изменяемый IP-адрес.

Наконец, для обеспечения безопасности сети многие сетевые устройства могут блокировать (фильтровать) пакеты, определенные поля которых имеют некоторые заранее заданные значения. Другими словами, при динамическом назначении адресов *усложняется фильтрация пакетов по IP-адресам.*

Последние две проблемы проще всего решаются отказом от динамического назначения адресов для интерфейсов, фигурирующих в системах мониторинга и безопасности.

ГЛАВА 14 Протокол межсетевого взаимодействия IP

IP-пакет

В каждой очередной сети, лежащей на пути перемещения пакета, протокол IP обращается к средствам транспортировки этой сети, чтобы с их помощью передать пакет на маршрутизатор, ведущий к следующей сети, или непосредственно на узел-получатель. *Поддержание интерфейса с нижележащими технологиями* подсетей является одной из важнейших функций протокола IP. В эти функции входит также *поддержание интерфейса с протоколами вышележащего транспортного уровня*, в частности с протоколом TCP, который решает все вопросы обеспечения надежной доставки данных по составной сети в стеке TCP/IP.

Протокол IP относится к протоколам *без установления соединений*, поддерживая обработку каждого IP-пакета как независимой единицы обмена, не связанной с другими пакетами. В протоколе IP нет механизмов, обычно применяемых для обеспечения достоверности конечных данных. Если во время продвижения пакета происходит какая-либо ошибка, то протокол IP по своей инициативе ничего не предпринимает для ее исправления. Например, если на промежуточном маршрутизаторе пакет был отброшен из-за ошибки по контрольной сумме, то модуль IP не пытается заново послать потерянный пакет. Другими словами, протокол IP реализует политику доставки «по возможности».

Имеется прямая связь между количеством полей заголовка пакета и функциональной сложностью протокола, который работает с этим заголовком. Чем проще заголовок, тем проще соответствующий протокол. Большая часть действий протокола связана с обработкой той служебной информации, которая переносится в полях заголовка пакета. Изучая назначение каждого поля заголовка IP-пакета (рис. 14.1), мы не только получаем формальные знания о структуре пакета, но и знакомимся с основными функциями протокола IP.

Поле **номера версии** занимает 4 бита и идентифицирует версию протокола IP. Сейчас повсеместно используется версия 4 (IPv4), хотя все чаще встречается и новая версия (IPv6). Значение **длины заголовка** IP-пакета также занимает 4 бита и измеряется в 32-битных словах. Обычно заголовок имеет длину в 20 байт (пять 32-битных слов), но при добавлении некоторой служебной информации это значение может быть увеличено за счет дополнительных байтов в поле параметров. Наибольшая длина заголовка составляет 60 байт.

Поле **типа сервиса** (Type of Service, ToS) имеет и другое, более современное название — **байт дифференцированного обслуживания**, или **DS-байт**. Этим двум названиям соответствуют два варианта интерпретации этого поля. В обоих случаях данное поле служит одной цели — хранению признаков, отражающих требования к качеству обслуживания пакета. В прежнем варианте первые три бита содержат значение **приоритета** пакета: от самого низкого — 0 до самого высокого — 7. Маршрутизаторы и компьютеры могут принимать во внимание приоритет пакета и обрабатывать более важные пакеты в первую очередь.



Рис. 14.1. Структура заголовка IP-пакета

Следующие три бита поля ToS определяют **критерий выбора маршрута**. Если для бита D (Delay — задержка) установлено значение 1, то маршрут должен выбираться для минимизации задержки доставки данного пакета, установленный бит T (Throughput — пропускная способность) — для максимизации пропускной способности, а бит R (Reliability — надежность) — для максимизации надежности доставки. Оставшиеся два бита имеют нулевое значение.

Стандарты дифференцированного обслуживания, принятые в конце 90-х, дали новое название этому полю, переопределив назначение его битов. В DS-байте также используются только старшие 6 бит, а два младших бита резервируются (о назначении битов DS-байта см. раздел «Поддержка QoS протоколом IP» в конце этой главы).

Поле **общей длины** занимает 2 байта и характеризует общую длину пакета с учетом заголовка и поля данных. Максимальная длина пакета ограничена разрядностью поля, определяющего эту величину, и составляет 65 535 байт, но в большинстве компьютеров и сетей столь большие пакеты не используются. При передаче по сетям различного типа длина пакета выбирается с учетом максимальной длины пакета протокола нижнего уровня, несущего IP-пакеты. Если это кадры Ethernet, то выбираются пакеты с максимальной длиной 1500 байт, умещающиеся в поле данных кадра Ethernet. В стандартах TCP/IP предусматривается, что все хосты должны быть готовы принимать пакеты длиной вплоть до 576 байт (независимо от того, приходят они целиком или фрагментами).

Идентификатор пакета занимает 2 байта и используется для распознавания пакетов, образовавшихся путем деления на части (фрагментации) исходного пакета. Все части (фрагменты) одного пакета должны иметь одинаковое значение этого поля.

Флаги занимают 3 бита и содержат признаки, связанные с фрагментацией. Установленный в 1 бит DF (Do not Fragment — не фрагментировать) запрещает маршрутизатору фрагментировать данный пакет, а установленный в 1 бит MF (More Fragments — больше фрагментов) говорит о том, что данный пакет является промежуточным (не последним) фрагментом. Оставшийся бит зарезервирован.

Поле **смещения фрагмента** занимает 13 бит и задает смещение в байтах поля данных этого фрагмента относительно начала поля данных исходного (нефрагментированного) пакета.

Используется при сборке/разборке фрагментов пакетов. Смещение должно быть кратно 8 байтам.

Поле **времени жизни** (Time To Live, TTL) занимает один байт и используется для задания предельного срока, в течение которого пакет может перемещаться по сети. Время жизни пакета измеряется в секундах и задается источником. По истечении каждой секунды пребывания на каждом из маршрутизаторов, через которые проходит пакет во время своего «путешествия» по сети, из его текущего времени жизни вычитается единица; единица вычитается и в том случае, если время пребывания — менее секунды. Поскольку современные маршрутизаторы редко обрабатывают пакет дольше, чем за одну секунду, то время жизни можно интерпретировать как максимальное число транзитных узлов, которые разрешено пройти пакету. Если значение поля времени жизни становится нулевым до того, как пакет достигает получателя, то пакет уничтожается. Таким образом, время жизни является своего рода часовым механизмом самоуничтожения пакета.

Поле **протокола верхнего уровня** занимает 1 байт и содержит идентификатор, указывающий, какому протоколу верхнего уровня принадлежит информация, размещенная в поле данных пакета. Значения идентификаторов для разных протоколов приводятся в документе RFC 1700, доступном по адресу http://www.iana.org. Например, 6 означает, что в пакете находится сообщение протокола TCP, 17 — протокола UDP, 1 — протокола ICMP.

Контрольная сумма заголовка занимает 2 байта (16 бит) и рассчитывается только по заголовку. Поскольку некоторые поля заголовка меняют свое значение в процессе передачи пакета по сети (например, поле времени жизни), контрольная сумма проверяется и повторно рассчитывается на каждом маршрутизаторе и конечном узле как дополнение к сумме всех 16-битных слов заголовка. При вычислении контрольной суммы значение самого поля контрольной суммы устанавливается в ноль. Если контрольная сумма неверна, то пакет отбрасывается, как только обнаруживается ошибка.

Поля **IP-адресов источника** и **приемника** имеют одинаковую длину — 32 бита.

Поле **параметров** является необязательным и используется обычно только при отладке сети. Это поле состоит из нескольких подполей одного из восьми предопределенных типов. В этих подполях можно указывать точный маршрут, по которому маршрутизаторы должны направлять данный пакет (то есть выполнять **маршрутизацию от источника**), регистрировать проходимые пакетом маршрутизаторы или помещать данные системы безопасности и временны́е отметки. Так как число подполей в поле параметров может быть произвольным, то в конце заголовка должно быть добавлено несколько нулевых байтов для **выравнивания** заголовка пакета по 32-битной границе.

Далее приведена распечатка значений полей заголовка одного из реальных IP-пакетов, захваченных в сети Ethernet средствами анализатора протоколов сетевого монитора (Network Monitor, NM) компании Microsoft. В данной распечатке NM в скобках дает шестнадцатеричные значения полей. Кроме того, программа иногда представляет числовые коды полей в виде, более удобном для чтения. Например, дружественный программный интерфейс NM интерпретирует код 6 в *поле протокола верхнего уровня*, помещая туда название соответствующего протокола — TCP (см. строку, выделенную полужирным шрифтом).

```
IP: Version = 4 (0x4)
IP: Header Length = 20 (0x14)
IP: Service Type = 0 (0x0)
```

```
IP: Precedence = Routine
IP: ...0.... = Normal Delay
IP: ....0... = Normal Throughput
IP: .....0.. = Normal Reliability
IP: Total Length = 54 (0x36)
IP: Identification = 31746 (0x7C02)
IP: Flags Summary = 2 (0x2)
IP: .......0 = Last fragment in datagram
IP: ......1. = Cannot fragment datagram
IP: Fragment Offset = 0 (0x0) bytes
IP: Time to Live = 128 (0x80)
IP: Protocol = TCP – Transmission Control
IP: Checksum = 0xEB86
IP: Source Address = 194.85.135.75
IP: Destination Address = 194.85.135.66
IP: Data: Number of data bytes remaining = 34 (0x0022)
```

Схема IP-маршрутизации

Рассмотрим механизм IP-маршрутизации на примере составной сети, представленной на рис. 14.2. В этой сети 20 маршрутизаторов (изображенных в виде пронумерованных квадрантных блоков) объединяют 18 сетей в общую сеть; N1, N2, ..., N18 — это номера сетей. На каждом маршрутизаторе и конечных узлах *A* и *B* функционируют протоколы IP.

К нескольким интерфейсам (портам) маршрутизаторов присоединяются сети. Каждый интерфейс маршрутизатора можно рассматривать как отдельный узел сети: он имеет сетевой адрес и локальный адрес в той подсети, которая к нему подключена. Например, маршрутизатор под номером 1 имеет три интерфейса, к которым подключены сети N1, N2, N3. На рисунке сетевые адреса этих портов обозначены IP_{11}, IP_{12} и IP_{13}. Интерфейс IP_{11} является узлом сети N1 и, следовательно, в поле номера сети порта IP_{11} содержится номер N1. Аналогично интерфейс IP_{12} — это узел в сети N2, а порт IP_{13} — узел в сети N3. Таким образом, маршрутизатор можно рассматривать как совокупность нескольких узлов, каждый из которых входит в свою сеть. Как единое устройство маршрутизатор не имеет выделенного адреса (ни сетевого, ни локального).

ПРИМЕЧАНИЕ

При наличии у маршрутизатора блока управления (например, по протоколу SNMP) этот блок имеет собственные локальный и сетевой адреса, по которым к нему обращается центральная станция управления. Эти адреса выбираются из того же пула, что и адреса физических интерфейсов маршрутизатора. В технической документации такого рода адреса называются адресами обратной петли (looopback address) или адресами виртуальных интерфейсов (virtual interface address). В отличие от адресов 127.x.x.x, зарезервированных для передачи данных между программными компонентами, находящимися в пределах одного компьютера, адреса виртуальных интерфейсов предполагают обращение к ним извне.

В сложных составных сетях почти всегда существуют несколько альтернативных маршрутов для передачи пакетов между двумя конечными узлами. Так, пакет, отправленный из узла *A* в узел *B*, может пройти через маршрутизаторы 17, 12, 5, 4 и 1 или маршрутизаторы 17, 13, 7, 6 и 3. Нетрудно найти еще несколько маршрутов между узлами *A* и *B*.

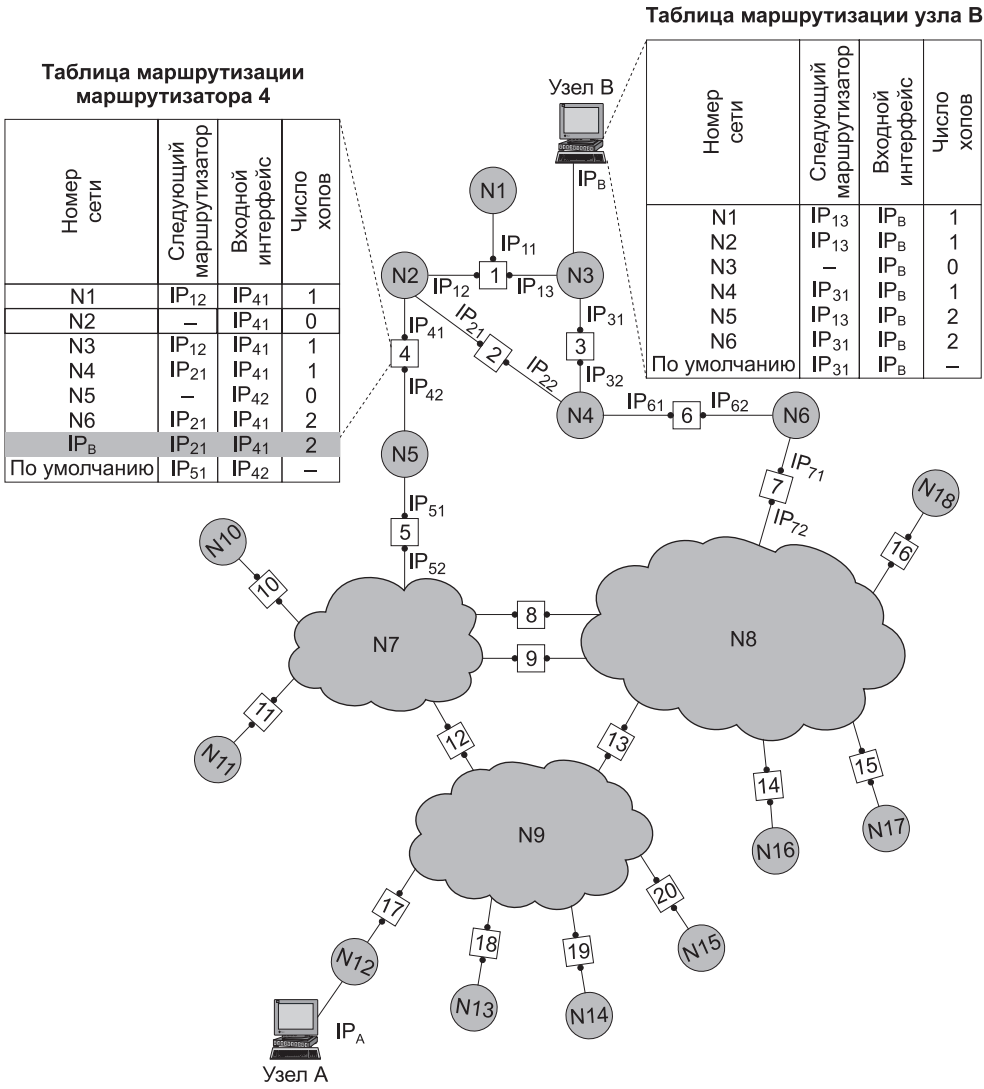


Рис. 14.2. Принципы маршрутизации в составной сети

Задачу выбора маршрута из нескольких возможных решают маршрутизаторы, а также конечные узлы. Маршрут выбирается на основании имеющейся у этих устройств информации о текущей конфигурации сети, а также на основании критерия выбора маршрута. В качестве критерия часто выступает задержка прохождения маршрута отдельным пакетом, средняя пропускная способность маршрута для последовательности пакетов или наиболее простой критерий, учитывающий только количество пройденных на маршруте промежуточных маршрутизаторов (*ретрансляционных участков*, или *хопов*). Полученная в результате анализа информация о маршрутах дальнейшего следования пакетов помещается в **таблицу маршрутизации**.

Упрощенная таблица маршрутизации

Используя условные обозначения для сетевых адресов маршрутизаторов и номеров сетей, показанные на рис. 14.2, посмотрим, как могла бы выглядеть таблица маршрутизации, например, в маршрутизаторе 4 (табл. 14.1).

Таблица 14.1. Таблица маршрутизации маршрутизатора 4

Адрес назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
N1	IP_{12} (R1)	IP_{41}	1
N2	—	IP_{41}	0 (подсоединена)
N3	IP_{12} (R1)	IP_{41}	1
N4	IP_{21} (R2)	IP_{41}	1
N5	—	IP_{42}	0 (подсоединена)
N6	IP_{21} (R2)	IP_{21}	2
IP_B	IP_{21} (R2)	IP_{41}	2
Маршрут по умолчанию	IP_{51} (R5)	IP_{42}	—

Первый столбец таблицы содержит **адреса назначения пакетов**.

В каждой строке таблицы следом за адресом назначения указывается **сетевой адрес следующего маршрутизатора** (точнее, сетевой адрес интерфейса следующего маршрутизатора), на который надо направить пакет, чтобы тот передвигался по направлению к заданному адресу по рациональному маршруту.

Перед тем как передать пакет следующему маршрутизатору, текущий маршрутизатор должен определить, на какой из нескольких собственных портов (IP_{41} или IP_{42}) он должен поместить данный пакет. Для этого служит третий столбец таблицы маршрутизации, содержащий **сетевые адреса выходных интерфейсов**.

Некоторые реализации сетевых протоколов допускают наличие в таблице маршрутизации сразу *нескольких строк*, соответствующих одному и тому же адресу назначения. В этом случае при выборе маршрута принимается во внимание столбец, представляющий расстояние до сети назначения. При этом расстояние измеряется в любой метрике, используемой в соответствии с заданным в сетевом пакете критерием. В табл. 14.1 расстояние между сетями измеряется хопами. Расстояние для сетей, непосредственно подключенных к портам маршрутизатора, здесь принимается равным 0, однако в некоторых реализациях отсчет расстояний начинается с 1.

Когда пакет поступает на маршрутизатор, модуль IP извлекает из его заголовка номер сети назначения и последовательно сравнивает его с номерами сетей из каждой строки таблицы. Строка с совпавшим номером сети показывает ближайший маршрутизатор, на который следует направить пакет. Например, если на какой-либо порт маршрутизатора 4 поступает пакет, адресованный в сеть N6, то из таблицы маршрутизации следует, что адрес следующего маршрутизатора — IP_{21}, то есть очередным этапом движения данного пакета будет движение к порту 1 маршрутизатора 2. Чаще всего в качестве адреса назначения

в таблице указывается не весь IP-адрес, а только номер сети назначения. Таким образом, для всех пакетов, направляемых в одну и ту же сеть, протокол IP будет предлагать один и тот же маршрут (мы пока не принимаем во внимание возможные изменения состояния сети — например, отказы маршрутизаторов или обрывы кабелей).

Однако в некоторых случаях возникает необходимость для одного из узлов сети определить **специфический маршрут**, отличающийся от маршрута, заданного для всех остальных узлов сети. Для этого в таблицу маршрутизации помещают для данного узла отдельную строку, содержащую его полный IP-адрес и соответствующую маршрутную информацию.

Такого рода запись имеется в табл. 14.1 для узла *B*. Предположим, администратор маршрутизатора 4, руководствуясь соображениями безопасности, решил направить пакеты, следующие в узел *B* (полный адрес IP_B), через маршрутизатор 2 (интерфейс IP_{21}), а не маршрутизатор 1 (интерфейс IP_{12}), через который передаются пакеты всем остальным узлам сети N3. Если в таблице имеются записи о маршрутах как к сети в целом, так и к ее отдельному узлу, то при поступлении пакета, адресованного данному узлу, маршрутизатор отдаст предпочтение специфическому маршруту.

Поскольку пакет может быть адресован *в любую сеть* составной сети, может показаться, что каждая таблица маршрутизации должна иметь записи обо *всех* сетях, входящих в составную сеть. Однако при таком подходе в случае крупной сети объем таблиц маршрутизации может оказаться очень большим, что повлияет на время ее просмотра, потребует много места для хранения и т. п. Поэтому на практике широко известен прием уменьшения количества записей в таблице маршрутизации, основанный на введении **маршрута по умолчанию** (default route), учитывающего особенности топологии сети. Рассмотрим, например, маршрутизаторы, находящиеся на периферии составной сети. В их таблицах достаточно записать номера только тех сетей, которые непосредственно подсоединены к данному маршрутизатору или расположены поблизости на тупиковых маршрутах. Обо всех остальных сетях можно сделать в таблице единственную запись, указывающую на маршрутизатор, через который пролегает путь ко всем этим сетям. Такой маршрутизатор называется **маршрутизатором по умолчанию** (default router). В нашем примере на маршрутизаторе 4 имеются специфические маршруты только для пакетов, следующих в сети N1–N6. Для всех остальных пакетов, адресованных в сети N7–N18, маршрутизатор предлагает продолжить путь через один и тот же порт IP_{51} маршрутизатора 5, который в данном случае и является маршрутизатором по умолчанию.

Таблицы маршрутизации конечных узлов

Задачу маршрутизации решают не только промежуточные узлы (маршрутизаторы), но и конечные узлы — компьютеры. Решение этой задачи начинается с того, что протокол IP на конечном узле определяет, направляется ли пакет в другую сеть или адресован какому-нибудь узлу данной сети. Если номер сети назначения совпадает с номером данной сети, то это означает, что пакет маршрутизировать не требуется. В противном случае маршрутизация нужна.

Структуры таблиц маршрутизации конечных узлов и транзитных маршрутизаторов аналогичны. Обратимся снова к сети, изображенной на рис. 14.2. Таблица маршрутизации конечного узла *B*, принадлежащего сети N3, могла бы выглядеть так, как табл. 14.2. Здесь

IP_B — сетевой адрес интерфейса компьютера *B*. На основании этой таблицы конечный узел *B* выбирает, на какой из двух имеющихся в локальной сети N3 маршрутизаторов (R1 или R3) следует посылать тот или иной пакет.

Таблица 14.2. Таблица маршрутизации конечного узла B

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
N1	IP_{13} (R1)	IP_B	1
N2	IP_{13} (R1)	IP_B	1
N3	—	IP_B	0
N4	IP_{31} (R3)	IP_B	1
N5	IP_{13} (R1)	IP_B	2
N6	IP_{31} (R3)	IP_B	2
Маршрут по умолчанию	IP_{31} (R3)	IP_B	—

Конечные узлы в еще большей степени, чем маршрутизаторы, пользуются приемом маршрутизации по умолчанию. Хотя они также в общем случае имеют в своем распоряжении таблицу маршрутизации, ее объем обычно незначителен, что объясняется периферийным расположением всех конечных узлов. Конечный узел часто вообще работает без таблицы маршрутизации, имея только сведения об адресе маршрутизатора по умолчанию. При наличии одного маршрутизатора в локальной сети этот вариант — единственно возможный для всех конечных узлов. Но даже при наличии нескольких маршрутизаторов в локальной сети, когда перед конечным узлом стоит проблема их выбора, часто в компьютерах для повышения производительности прибегают к заданию маршрута по умолчанию.

Рассмотрим таблицу маршрутизации другого конечного узла составной сети — узла *A* (табл. 14.3). Компактный вид таблицы маршрутизации узла *A* отражает тот факт, что все пакеты, направляемые из узла *A*, либо не выходят за пределы сети N12, либо непременно проходят через порт 1 маршрутизатора 17. Этот маршрутизатор и определен в таблице маршрутизации в качестве маршрутизатора по умолчанию.

Таблица 14.3. Таблица маршрутизации конечного узла A

Номер сети назначения	Сетевой адрес следующего маршрутизатора	Сетевой адрес выходного порта	Расстояние до сети назначения
N12	—	IPA	0
Маршрут по умолчанию	$IP_{17,1}$ (R17)	IPA	—

Еще одним отличием работы маршрутизатора и конечного узла является способ построения таблицы маршрутизации. Если маршрутизаторы, как правило, автоматически создают таблицы маршрутизации, обмениваясь служебной информацией, то для конечных узлов таблицы маршрутизации часто создаются вручную администраторами и хранятся в виде постоянных файлов на дисках.

Просмотр таблиц маршрутизации без масок

Рассмотрим алгоритм просмотра таблицы маршрутизации протоколом IP, используя данные из табл. 14.1 и рис. 14.2. Пусть на один из интерфейсов маршрутизатора поступает пакет. Протокол IP извлекает из пакета IP-адрес назначения (предположим, адрес назначения IP_B).

1. Выполняется первая фаза просмотра таблицы — поиск специфического маршрута к узлу. IP-адрес (целиком) последовательно, строка за строкой, сравнивается с содержимым поля адреса назначения таблицы маршрутизации. Если произошло совпадение (как в табл. 14.1), то из соответствующей строки извлекаются адрес следующего маршрутизатора (IP_{21}) и идентификатор выходного интерфейса (IP_{41}). На этом просмотр таблицы заканчивается.
2. Предположим теперь, что в таблице нет строки с адресом назначения IPB, а значит, совпадения не произошло. В этом случае протокол IP переходит ко второй фазе просмотра — поиску маршрута к сети назначения. Из IP-адреса выделяется номер сети (в нашем примере из адреса IP_B выделяется номер сети N3), после чего таблица снова просматривается на предмет совпадения номера сети в какой-либо строке с номером сети из пакета. При совпадении (а в нашем примере оно произошло) из соответствующей строки таблицы извлекаются адрес следующего маршрутизатора (IP_{12}) и идентификатор выходного интерфейса (IP_{41}). Просмотр таблицы на этом завершается.
3. Наконец, предположим, что адрес назначения в пакете был таков, что совпадения не произошло ни в первой, ни во второй фазах просмотра. В таком случае средствами протокола IP либо выбирается маршрут по умолчанию (и пакет направляется по адресу IP_{51}), либо, если маршрут по умолчанию отсутствует, пакет отбрасывается[1]. Просмотр таблицы на этом заканчивается.

ВНИМАНИЕ

Последовательность фаз в данном алгоритме строго определена, в то время как последовательность просмотра или, что одно и то же, порядок расположения строк в таблице, включая запись о маршруте по умолчанию, никак не сказывается на результате.

Примеры таблиц маршрутизации разных форматов

Структура реальных таблиц маршрутизации стека TCP/IP в целом соответствует упрощенной структуре рассмотренных ранее таблиц. Отметим, однако, что вид таблицы IP-маршрутизации зависит от конкретной реализации стека TCP/IP. Приведем пример нескольких вариантов таблицы маршрутизации, с которыми мог бы работать маршрутизатор R1 в сети, представленной на рис. 14.3.

Начнем с придуманного, предельно упрощенного варианта таблицы маршрутизации (табл. 14.4). Здесь имеются три маршрута к сетям (записи 56.0.0.0, 116.0.0.0 и 129.13.0.0), две записи о непосредственно подсоединенных сетях (198.21.17.0 и 213.34.12.0), а также запись о маршруте по умолчанию.

[1] Стандарты технологии TCP/IP не требуют, чтобы в таблице маршрутизации непременно содержались маршруты для всех пакетов, которые могут прийти на его интерфейсы, более того, в таблице может отсутствовать маршрут по умолчанию.

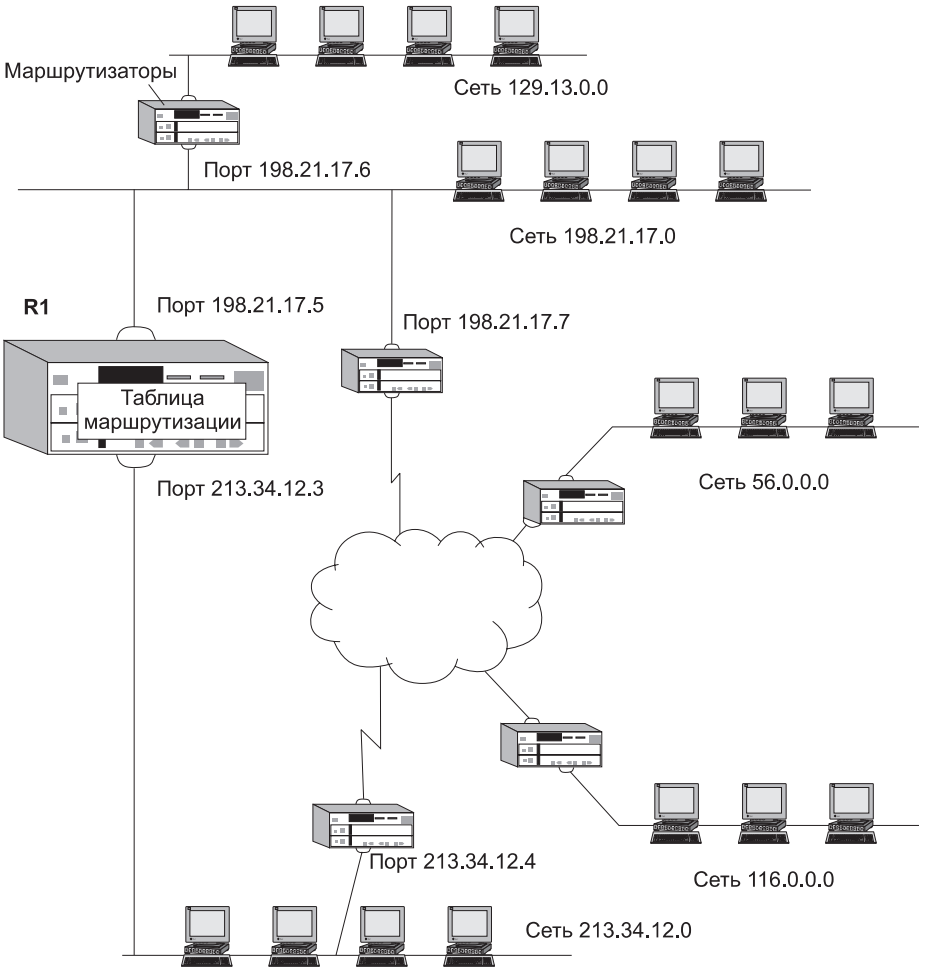


Рис. 14.3. Пример маршрутизируемой сети

Таблица 14.4. Упрощенная таблица маршрутизации маршрутизатора R1

Адрес сети назначения	Адрес следующего маршрутизатора	Адрес выходного интерфейса	Расстояние до сети назначения
56.0.0.0	213.34.12.4	213.34.12.3	15
116.0.0.0	213.34.12.4	213.34.12.3	13
129.13.0.0	198.21.17.6	198.21.17.5	2
198.21.17.0	198.21.17.5	198.21.17.5	1 (подсоединена)
213.34.12.0	213.34.12.3	213.34.12.3	1 (подсоединена)
Маршрут по умолчанию	198.21.17.7	198.21.17.5	—

Более сложный вид имеют таблицы, которые генерируются в промышленно выпускаемом сетевом оборудовании. Если представить, что в качестве маршрутизатора R1 в данной сети работает штатный *программный* маршрутизатор ОС Windows, то его таблица маршрутизации могла бы выглядеть, как табл. 14.5.

Таблица 14.5. Таблица программного маршрутизатора ОС Windows

Сетевой адрес	Маска	Адрес шлюза	Интерфейс	Метрика
127.0.0.0	255.0.0.0	127.0.0.1	127.0.0.1	1
0.0.0.0	0.0.0.0	198.21.17.7	198.21.17.5	1
56.0.0.0	255.0.0.0	213.34.12.4	213.34.12.3	15
116.0.0.0	255.0.0.0	213.34.12.4	213.34.12.3	13
129.13.0.0	255.255.0.0	198.21.17.6	198.21.17.5	2
198.21.17.0	255.255.255.0	198.21.17.5	198.21.17.5	1
198.21.17.5	255.255.255.255	127.0.0.1	127.0.0.1	1
198.21.17.255	255.255.255.255	198.21.17.5	198.21.17.5	1
213.34.12.0	255.255.255.0	213.34.12.3	213.34.12.3	1
213.34.12.3	255.255.255.255	127.0.0.1	127.0.0.1	1
213.34.12.255	255.255.255.255	213.34.12.3	213.34.12.3	1
224.0.0.0	224.0.0.0	198.21.17.6	198.21.17.6	1
224.0.0.0	224.0.0.0	213.34.12.3	213.34.12.3	1
255.255.255.255	255.255.255.255	198.21.17.6	198.21.17.6	1

Если на месте маршрутизатора R1 установить один из популярных *аппаратных* маршрутизаторов, то его таблица маршрутизации для этой же сети может выглядеть совсем иначе (табл. 14.6).

Таблица 14.6. Таблица маршрутизации аппаратного маршрутизатора

Адрес назначения	Маска	Шлюз	Метрика	Статус	TTL	Источник
198.21.17.0	255.255.255.0	198.21.17.5	0	Up	—	Подключена
213.34.12.0	255.255.255.0	213.34.12.3	0	Up	—	Подключена
56.0.0.0	255.0.0.0	213.34.12.4	14	Up	—	Статическая
116.0.0.0	255.0.0.0	213.34.12.4	12	Up	—	Статическая
129.13.0.0	255.255.0.0	198.21.17.6	1	Up	160	RIP

И наконец, табл. 14.7 представляет собой таблицу маршрутизации для того же маршрутизатора R1, реализованного в виде программного маршрутизатора одной из версий ОС Unix.

Таблица 14.7. Таблица маршрутизации маршрутизатора Unix

Адрес назначения	Шлюз	Флаги	Число ссылок	Загрузка	Интерфейс
127.0.0.0	127.0.0.1	UH	1	154	lo0
Маршрут по умолчанию	198.21.17.7	UG	5	43270	le0
198.21.17.0	198.21.17.5	U	35	246876	le0
213.34.12.0	213.34.12.3	U	44	132435	le1
129.13.0.0	198.21.1.7.6	UG	6	16450	le0
56.0.0.0	213.34.12.4	UG	12	5764	le1
116.0.0.0	213.34.12.4	UG	21	23544	le1

ПРИМЕЧАНИЕ

Заметим, что поскольку между структурой сети и таблицей маршрутизации нет однозначного соответствия, для каждого из приведенных вариантов таблицы можно предложить свои «подварианты», отличающиеся выбранным маршрутом к той или иной сети. В данном случае внимание концентрируется на существенных различиях в форме представления маршрутной информации разными реализациями маршрутизаторов.

Несмотря на достаточно заметные внешние различия, в каждой из трех «реальных» таблиц присутствуют все ключевые данные из рассмотренной упрощенной таблицы, без которых невозможна маршрутизация пакетов.

К таким данным, во-первых, относятся *адреса сети назначения* (столбцы «Адрес назначения» в аппаратном маршрутизаторе и маршрутизаторе ОС Unix или столбец «Сетевой адрес» в маршрутизаторе ОС Windows).

Вторым обязательным полем таблицы маршрутизации является *адрес следующего маршрутизатора* (столбец «Шлюз» в аппаратном маршрутизаторе и маршрутизаторе ОС Unix или столбец «Адрес шлюза» в маршрутизаторе ОС Windows).

Третий ключевой параметр — *адрес порта*, на который нужно направить пакет, в некоторых таблицах указывается прямо (столбец «Интерфейс» в таблице маршрутизатора ОС Windows), в некоторых — косвенно. Так, в таблице маршрутизатора ОС Unix вместо адреса порта задается его условное наименование: le0 для порта с адресом 198.21.17.5, le1 для порта с адресом 213.34.12.3 и lo0 для внутреннего порта с адресом 127.0.0.1. В аппаратном маршрутизаторе поле, обозначающее выходной порт в какой-либо форме, вообще отсутствует. Это объясняется тем, что адрес выходного порта всегда можно косвенно определить по адресу следующего маршрутизатора. Например, определим по табл. 14.6 адрес выходного порта для сети 56.0.0.0. Из таблицы следует, что следующим маршрутизатором для этой сети будет маршрутизатор с адресом 213.34.12.4. Адрес следующего маршрутизатора должен принадлежать одной из непосредственно присоединенных к маршрутизатору сетей, и в данном случае — это сеть 213.34.12.0. Маршрутизатор имеет порт, присоединенный к этой сети, адрес которого (213.34.12.3) мы находим в столбце «Шлюз» второй строки таблицы маршрутизации, описывающей непосредственно присоединенную сеть 213.34.12.0. Для непосредственно присоединенных сетей адресом следующего маршрутизатора всегда

является адрес собственного порта маршрутизатора. Таким образом, для сети 56.0.0 адресом выходного порта является 213.34.12.3.

Стандартным решением сегодня является использование поля маски в каждой записи таблицы, как это сделано в таблицах маршрутизатора ОС Windows и аппаратного маршрутизатора (столбцы «Маска»). Механизм обработки масок при принятии решения маршрутизаторами рассматривается далее. Отсутствие поля маски говорит о том, что либо маршрутизатор рассчитан на работу только с тремя стандартными классами адресов, либо для всех записей используется одна и та же маска, что снижает гибкость маршрутизации.

Поскольку в таблице маршрутизации маршрутизатора ОС Unix каждая сеть назначения упомянута только один раз, а значит, возможность выбора маршрута отсутствует, то поле метрики является необязательным параметром. В остальных двух таблицах поле метрики используется только для указания на то, что сеть подключена непосредственно. Метрика 0 для аппаратного маршрутизатора или 1 для маршрутизатора ОС Windows говорит маршрутизатору, что эта сеть непосредственно подключена к его порту, а другое значение метрики соответствует удаленной сети. Выбор метрики для непосредственно подключенной сети (1 или 0) является произвольным — главное, чтобы метрика удаленной сети отсчитывалась с учетом этого выбранного начального значения. В маршрутизаторе Unix используется поле признаков, где флаг G (Gateway — шлюз) отмечает удаленную сеть, а его отсутствие — непосредственно подключенную.

Признак непосредственно подключенной сети говорит маршрутизатору, что пакет уже достиг своей сети, поэтому протокол IP активизирует ARP-запрос относительно IP-адреса *узла назначения*, а не следующего маршрутизатора.

Однако существуют ситуации, когда маршрутизатор должен обязательно хранить значение метрики для записи о каждой удаленной сети. Эти ситуации возникают, когда записи в таблице маршрутизации являются результатом работы некоторых протоколов маршрутизации, например протокола RIP. В таких протоколах новая информация о какой-либо удаленной сети сравнивается с информацией, содержащейся в таблице в данный момент, и, если значение новой метрики лучше текущей, то новая запись вытесняет имеющуюся. В таблице маршрутизатора ОС Unix поле метрики отсутствует, и это значит, что он не использует протокол RIP.

Флаги записей присутствуют только в таблице маршрутизатора ОС Unix:

- U — маршрут активен и работоспособен. Аналогичный смысл имеет поле статуса в аппаратном маршрутизаторе.
- H — признак специфического маршрута к определенному хосту.
- G — маршрут пакета проходит через промежуточный маршрутизатор (шлюз). Отсутствие этого флага отмечает непосредственно подключенную сеть.
- D — маршрут получен из перенаправленного сообщения протокола ICMP. Этот признак может присутствовать только в таблице маршрутизации *конечного узла*. Признак означает, что конечный узел при какой-то предыдущей передаче пакета выбрал не самый рациональный следующий маршрутизатор на пути к данной сети, а этот маршрутизатор с помощью протокола ICMP сообщил конечному узлу, что все последующие пакеты к данной сети нужно отправлять через другой маршрутизатор.

В таблице маршрутизатора ОС Unix используются еще два поля, имеющих справочное значение. Поле числа ссылок показывает, сколько раз на данный маршрут ссылались при продвижении пакетов. Поле загрузки отражает количество байтов, переданных по данному маршруту.

В записях таблиц аппаратного маршрутизатора также имеются два справочных поля. Поле **времени жизни записи** (TTL) в данном случае никак не связано со временем жизни пакета. Здесь оно показывает время, в течение которого значение данной записи еще действительно. **Поле источника** говорит об источнике появления записи в таблице маршрутизации.

Источники и типы записей в таблице маршрутизации

Практически для всех маршрутизаторов существуют *три* основных источника записей в таблице.

Во-первых, это **программное обеспечение** стека TCP/IP, которое при инициализации маршрутизатора автоматически заносит в таблицу несколько записей, в результате чего создается так называемая *минимальная таблица маршрутизации*. Программное обеспечение формирует записи о непосредственно подключенных сетях и маршрутах по умолчанию, информация о которых появляется в стеке при ручном конфигурировании интерфейсов компьютера или маршрутизатора. К таким записям в приведенных примерах относятся записи о сетях 213.34.12.0 и 198.21.17.0, запись о маршруте по умолчанию в маршрутизаторе ОС Unix и запись 0.0.0.0 в маршрутизаторе ОС Windows. Кроме того, программное обеспечение автоматически заносит в таблицу маршрутизации записи об адресах особого назначения. В приведенных примерах таблица маршрутизатора ОС Windows содержит наиболее полный набор записей такого рода. Несколько записей в этой таблице связано с особым адресом 127.0.0.0. Записи с адресом 224.0.0.0 требуются для обработки групповых адресов. Кроме того, в таблицу могут быть занесены адреса, предназначенные для обработки широковещательных рассылок (например, записи 8 и 11 содержат адрес отправки широковещательного сообщения в соответствующих подсетях, а последняя запись в таблице — адрес ограниченной широковещательной рассылки). Заметим, в некоторых таблицах записи об особых адресах вообще отсутствуют.

Во-вторых, источником записей в таблице является **администратор**, непосредственно формирующий записи с помощью некоторой системной утилиты, например программы *route*, доступной в ОС Unix и ОС Windows. В аппаратных маршрутизаторах также всегда имеется команда для ручного задания записей таблицы маршрутизации. Заданные вручную записи всегда являются *статическими*, то есть не имеют срока жизни. Эти записи могут быть как постоянными, то есть сохраняющимися при перезагрузке маршрутизатора, так и временными, хранящимися в таблице только до выключения устройства. Часто администратор вручную заносит запись о маршруте по умолчанию. Таким же образом в таблицу маршрутизации может быть внесена и запись о специфическом для узла маршруте.

И наконец, третьим источником записей могут быть **протоколы маршрутизации**, такие как RIP или OSPF. Эти записи всегда являются *динамическими*, то есть имеют ограниченный срок жизни.

Программные маршрутизаторы Windows и Unix не показывают источник появления той или иной записи в таблице, а аппаратный маршрутизатор использует для этой цели поле

источника. В приведенном в табл. 14.6 примере первые две записи созданы программным обеспечением стека на основании данных о конфигурации портов маршрутизатора — это показывает признак «Подключена». Следующие две записи обозначены как статические — это означает, что их ввел вручную администратор. Последняя запись — следствие работы протокола RIP, поэтому в ее поле «TTL» имеется значение 160.

Пример IP-маршрутизации без масок

Рассмотрим процесс продвижения пакета в составной сети на примере IP-сети, показанной на рис. 14.4, исходя из того, что все узлы сети, рассматриваемой в примере, имеют *адреса, основанные на классах*. Особое внимание уделим взаимодействию протокола IP с протоколами разрешения адресов ARP и DNS.

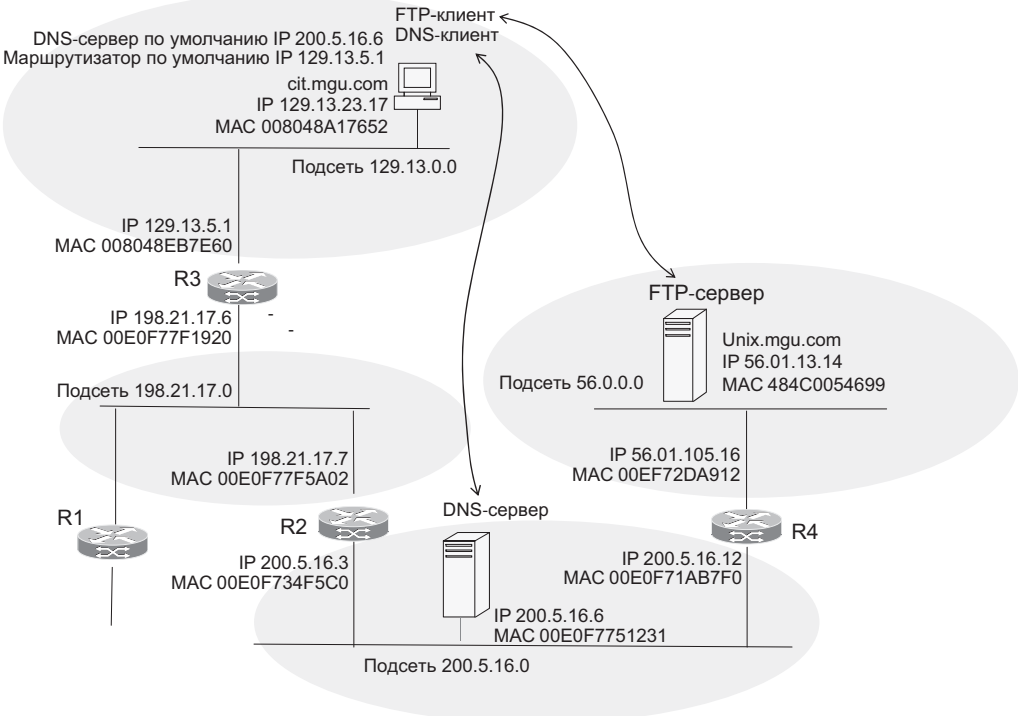


Рис. 14.4. Пример IP-маршрутизации

Итак, пусть пользователю компьютера cit.mgu.com, находящегося в сети 129.13.0.0, необходимо установить связь с FTP-сервером. Пользователю известно символьное имя сервера unix.mgu.com, поэтому он набирает на клавиатуре команду обращения к FTP-серверу по имени:

```
> ftp unix.mgu.com
```

Выполнение этой команды инициирует три последовательные операции:

1. DNS-клиент (работающий на компьютере cit.mgu.com) передает DNS-серверу сообщение, в котором содержится запрос об IP-адресе сервера unix.mgu.com, с которым он хочет связаться по протоколу FTP.
2. DNS-сервер, выполнив поиск, передает ответ DNS-клиенту о найденном IP-адресе сервера unix.mgu.com.
3. FTP-клиент (работающий на том же компьютере cit.mgu.com), используя найденный IP-адрес сервера unix.mgu.com, передает ему сообщение.

Давайте последовательно, по шагам, рассмотрим, как при решении этих задач взаимодействуют между собой протоколы DNS, IP, ARP и Ethernet и что происходит при этом с кадрами и пакетами.

Формирование IP-пакета с инкапсулированным в него DNS-запросом. Программный модуль FTP-клиента, получив команду > ftp unix.mgu.com, делает запрос к работающему на этом же компьютере DNS-клиенту, который формирует запрос к DNS-серверу: «Какой IP-адрес соответствует имени unix.mgu.com?» Запрос упаковывается в UDP-дейтаграмму, затем в IP-пакет. В заголовке пакета в качестве адреса назначения указывается IP-адрес 200.5.16.6 DNS-сервера, который входит в число конфигурационных параметров хоста.

Передача кадра Ethernet с IP-пакетом маршрутизатору R3. Для передачи этого IP-пакета необходимо его упаковать в кадр Ethernet, указав в заголовке MAC-адрес получателя. Технология Ethernet способна доставлять кадры только тем адресатам, которые находятся в пределах одной подсети с отправителем. Если же адресат расположен вне этой подсети, то кадр надо передать ближайшему маршрутизатору, чтобы тот взял на себя заботу о дальнейшем перемещении пакета. Модуль IP, сравнив номера сетей в адресах отправителя и получателя, то есть 129.13.23.17 и 200.5.16.6, выясняет, что пакет направляется в другую сеть, следовательно, его необходимо передать маршрутизатору, в данном случае маршрутизатору по умолчанию. IP-адрес маршрутизатора по умолчанию также известен клиентскому узлу, поскольку он входит в число конфигурационных параметров. Однако в кадре Ethernet необходимо указать не IP-адрес, а MAC-адрес получателя. Эта проблема решается с помощью протокола ARP, который для ответа на вопрос «Какой MAC-адрес соответствует IP-адресу 129.13.5.1?» делает поиск в своей ARP-таблице. Поскольку обращения к маршрутизатору происходят часто, будем считать, что нужный MAC-адрес обнаруживается в таблице и имеет значение 008048EB7E60. После получения этой информации клиентский компьютер cit.mgu.com отправляет маршрутизатору R3 кадр Ethernet с DNS-запросом (рис. 14.5).

Определение IP-адреса и MAC-адреса следующего маршрутизатора R2. Кадр принимается интерфейсом 129.13.5.1 маршрутизатора R3. Протокол Ethernet, работающий на этом интерфейсе, извлекает из этого кадра IP-пакет и передает его протоколу IP. Протокол IP находит в заголовке пакета адрес назначения 200.5.16.6 и просматривает записи своей таблицы маршрутизации. Пусть маршрутизатор R3 не обнаруживает специфического маршрута для адреса назначения 200.5.16.6, но находит в своей таблице следующую запись:

```
200.5.16.0 198.21.17.7 198.21.17.6
```

Эта запись говорит о том, что пакеты для сети 200.5.16.0 маршрутизатор R3 должен передавать на свой выходной интерфейс 198.21.17.6, с которого они поступят на интерфейс следующего маршрутизатора R2, имеющего IP-адрес 198.21.17.7. Далее протокол ARP определяет MAC-адрес маршрутизатора R2. Пусть на этот раз в ARP-таблице нет записи

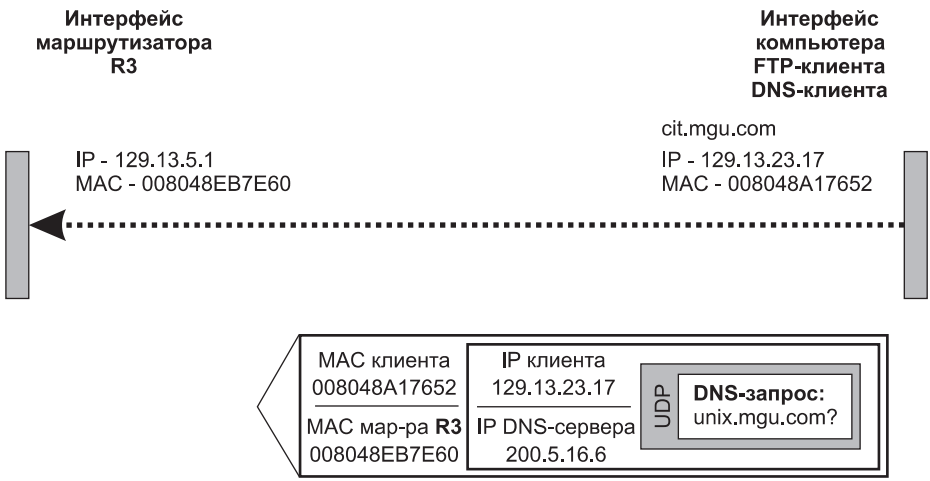


Рис. 14.5. Кадр Ethernet с инкапсулированным IP-пакетом, отправленный с клиентского компьютера

об адресе маршрутизатора R2. Тогда в сеть отправляется широковещательный ARP-запрос, поступающий на все интерфейсы сети 198.21.17.0. Ответ приходит только от интерфейса маршрутизатора R2: «Я имею IP-адрес 198.21.17.7, и мой MAC-адрес 00E0F77F5A02». Зная MAC-адрес маршрутизатора R2 (00E0F77F5A02), маршрутизатор R3 передает ему IP-пакет с DNS-запросом (рис. 14.6).

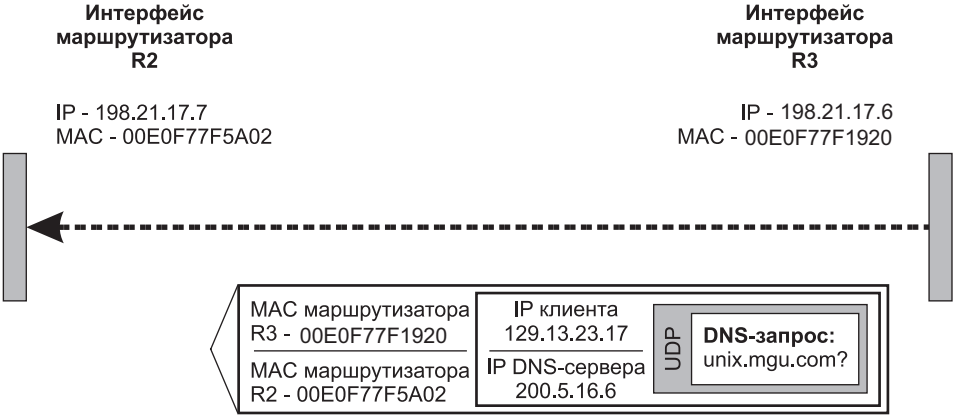


Рис. 14.6. Кадр Ethernet с DNS-запросом, отправленный с маршрутизатора R3 маршрутизатору R2

Маршрутизатор R2 доставляет пакет DNS-серверу. Для этого модуль IP на маршрутизаторе R2 извлекает из пакета IP-адрес назначения и, просматривая свою таблицу маршрутизации, обнаруживает, что сеть назначения 200.5.16.0 является непосредственно присоединенной к его второму интерфейсу. Следовательно, пакет не нужно маршрутизировать, однако требуется определить MAC-адрес узла назначения. Протокол ARP «по

просьбе» протокола IP находит (либо в ARP-таблице, либо широковещательно) MAC-адрес 00E0F7751231 DNS-сервера. Маршрутизатор R2 формирует кадр Ethernet с DNS-запросом (рис. 14.7) и передает его в сеть.

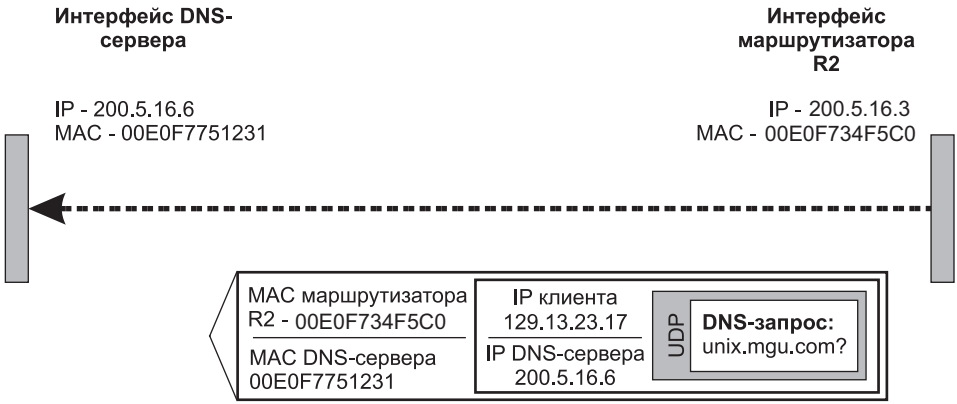


Рис. 14.7. Кадр Ethernet с DNS-запросом, отправленный с маршрутизатора R2

Сетевой адаптер DNS-сервера захватывает кадр Ethernet, обнаруживает совпадение MAC-адреса назначения, содержащегося в заголовке, со своим собственным адресом и направляет его вышележащим протоколам. DNS-запрос передается программному модулю DNS-сервера. DNS-сервер просматривает свои таблицы, возможно, обращается к другим DNS-серверам и в результате формирует ответ: «Символьному имени unix.mgu.com соответствует IP-адрес 56.01.13.14».

Процесс доставки DNS-ответа (рис. 14.8) клиенту cit.mgu.com аналогичен процессу передачи DNS-запроса, который мы только что так подробно описали. FTP-клиент, получив IP-адрес FTP-сервера, посылает ему свое сообщение. Для читателя будет весьма полезно воспроизвести процесс перемещения этого сообщения по сети, обращая особое внимание на значения адресных полей кадров и пакетов.

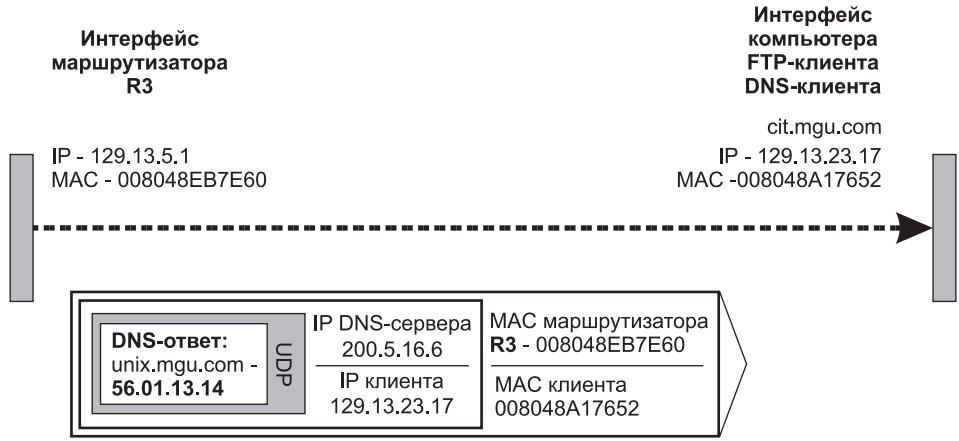


Рис. 14.8. Кадр Ethernet с DNS-ответом, отправленный с маршрутизатора R3 компьютеру-клиенту

ПРИМЕЧАНИЕ

Заметим, что во время всего путешествия пакета по составной сети от клиентского компьютера до DNS-сервера IP-адреса получателя и отправителя в полях заголовка IP-пакета не изменяются. Зато MAC-адреса отправителя и получателя изменяются на каждом отрезке пути.

Маршрутизация с использованием масок

Алгоритм маршрутизации усложняется, когда в систему адресации узлов вносятся дополнительные элементы — маски. Часто администраторы сетей испытывают неудобства, поскольку количества централизованно выделенных им номеров сетей недостаточно для того, чтобы структурировать сеть надлежащим образом, например, развести все слабо взаимодействующие компьютеры по разным сетям. В такой ситуации возможны два пути. Первый — получение от какого-либо центрального органа дополнительных номеров сетей. Второй, более распространенный, — использование технологии масок, позволяющей разделить одну имеющуюся сеть на несколько.

Структуризация сети масками одинаковой длины

С о сложностями использования масок администратор впервые сталкивается н е тогда, когда начинает конфигурировать сетевые интерфейсы и создавать таблицы маршрутизации, а гораздо раньше — на этапе планирования сети. Планирование включает в себя определение количества сетей, из которых будет состоять корпоративная сеть, оценку требуемого количества адресов для каждой сети, получение пула адресов от поставщика услуг, распределение адресного пространства между сетями. Последняя задача часто оказывается нетривиальной, особенно когда решается в условиях дефицита адресов.

Допустим, администратор получил в свое распоряжение сеть класса B: 129.44.0.0. Он может организовать сеть с большим числом узлов, номера которых доступны ему из диапазона 0.0.0.1–0.0.255.254. Всего в его распоряжении имеется ($2^{16} - 2$) адреса (вычитание двойки связано с тем, что, как уже отмечалось, адреса из одних нулей и одних единиц имеют специальное назначение и не годятся для адресации узлов). Однако ему не нужна одна большая неструктурированная сеть. Производственная необходимость диктует администратору другое решение, в соответствии с которым сеть должна быть разделена на три отдельных подсети, при этом трафик в каждой подсети должен быть надежно локализован. Это позволит легче диагностировать сеть и проводить в каждой из подсетей особую политику безопасности. Заметим, что разделение большой сети с помощью масок имеет еще одно преимущество — оно позволяет скрыть внутреннюю структуру сети предприятия от внешнего наблюдения и тем самым повысить ее безопасность.

На рис. 14.9 показано разделение всего полученного администратором адресного диапазона на четыре равные части — каждая по 2^{14} адресов. При этом число разрядов, доступное для нумерации узлов, ***уменьшилось*** на два бита, а префикс (номер) каждой из четырех сетей стал ***длиннее*** на два бита. Следовательно, каждый из четырех диапазонов можно записать в виде IP-адреса с маской, состоящей из 18 единиц, или в десятичной нотации — 255.255.192.0.

129.44.0.0/18 (10000001 00101100 **00**000000 00000000)
129.44.64.0/18 (10000001 00101100 **01**000000 00000000)
129.44.128.0/18 (10000001 00101100 **10**000000 00000000)
129.44.192.0/18 (10000001 00101100 **11**000000 00000000)

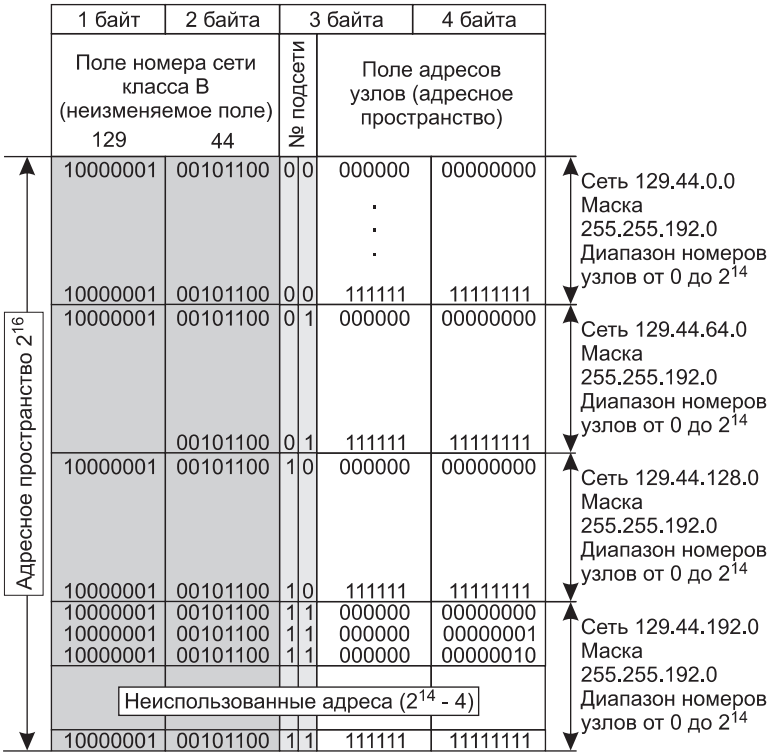


Рис. 14.9. Разделение адресного пространства 129.44.0.0 сети класса B на четыре равные части

Из записей видно, что администратор получает возможность использовать для нумерации подсетей два дополнительных бита (полужирный шрифт), что позволяет выделить из одной централизованно выделенной сети четыре, в данном примере — сети 129.44.0.0/18, 129.44.64.0/18, 129.44.128.0/18, 129.44.192.0/18 (рис. 14.10).

ПРИМЕЧАНИЕ

Некоторые программные и аппаратные маршрутизаторы, следуя устаревшим рекомендациям RFC 950, не поддерживают номера подсетей, которые состоят либо только из одних нулей, либо только из одних единиц. Например, для такого типа оборудования номер сети 129.44.0.0 с маской 255.255.192.0, использованной в нашем примере, окажется недопустимым, поскольку в этом случае разряды в поле номера подсети имеют значение 00. По аналогичным соображениям недопустимым может оказаться номер сети 129.44.192.0 с тем же значением маски. Здесь номер подсети состоит только из единиц. Однако современные маршрутизаторы, поддерживающие рекомендации RFC 1878, свободны от этих ограничений.

Весь трафик во внутреннюю сеть 129.44.0.0, направляемый из внешней сети, поступает через маршрутизатор R1. В целях структуризации информационных потоков во внутренней сети установлен дополнительный маршрутизатор R2. Каждая из вновь образованных сетей подключена к соответственно сконфигурированным портам внутреннего маршрутизатора R2.

ПРИМЕЧАНИЕ

В одной из этих сетей (129.44.192.0/18), выделенной для организации соединения между внешним и внутренним маршрутизаторами, для адресации узлов задействованы всего два адреса — 129.44.192.1 (порт маршрутизатора R2) и 129.44.192.2 (порт маршрутизатора R1). Огромное число узлов в этой подсети не используется. Такой пример выбран исключительно в учебных целях, чтобы показать неэффективность сетей равного размера.

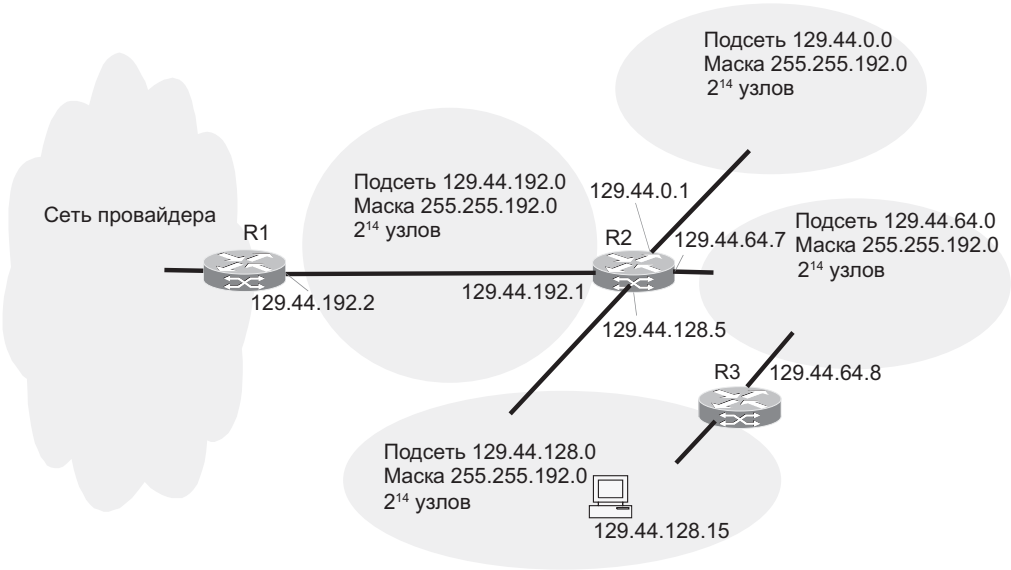


Рис. 14.10. Маршрутизация с использованием масок одинаковой длины

Извне, со стороны маршрутизатора R1, сеть по-прежнему выглядит как единая сеть класса B. Но поступающий в сеть общий трафик разделяется локальным маршрутизатором R2 между четырьмя сетями. В условиях, когда механизм классов не действует, маршрутизатор должен иметь другое средство, которое позволило бы ему определять, какая часть 32-битного числа, помещенного в поле адреса назначения, является номером сети. Именно этой цели служит дополнительное поле маски, включенное в таблицу маршрутизации (табл. 14.8).

Первые четыре записи в таблице относятся к внутренним подсетям, непосредственно подключенным к портам маршрутизатора R2. Запись 0.0.0.0 с маской 0.0.0.0 соответствует маршруту по умолчанию. Последняя запись определяет специфический маршрут к узлу 129.44.128.15. В тех строках таблицы, в которых в качестве адреса назначения указан пол-

ный IP-адрес узла, маска имеет значение 255.255.255.255. В отличие от всех других узлов сети 129.44.128.0, к которым пакеты поступают с интерфейса 129.44.128.5 маршрутизатора R2, к данному узлу они должны приходить через маршрутизатор R3.

Таблица 14.8. Таблица маршрутизатора R2 в сети с масками одинаковой длины

Адрес назначения	Маска	Адрес следующего маршрутизатора	Адрес порта	Расстояние
129.44.0.0	255.255.192.0	129.44.0.1	129.44.0.1	Подключена
129.44.64.0	255.255.192.0	129.44.64.7	129.44.64.7	Подключена
129.44.128.0	255.255.192.0	129.44.128.5	129.44.128.5	Подключена
129.44.192.0	255.255.192.0	129.44.192.1	129.44.192.1	Подключена
0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	—
129.44.128.15	255.255.255.255	129.44.64.8	129.44.64.7	—

Просмотр таблиц маршрутизации с учетом масок

Алгоритм просмотра таблиц маршрутизации с учетом масок, во многом подобный алгоритму просмотра таблиц без масок, имеет и ряд существенных отличий. Поиск следующего маршрутизатора для вновь поступившего IP-пакета протокол начинает с того, что *извлекает из пакета адрес назначения* (обозначим его IP_D). Затем протокол IP приступает к процедуре просмотра таблицы маршрутизации, также состоящей из двух фаз, как и процедура просмотра таблицы, в которой столбец маски отсутствует.

1. *Первая фаза* состоит в *поиске специфического маршрута* для адреса IP_D. С этой целью из каждой записи таблицы, в которой маска имеет значение 255.255.255.255, извлекается адрес назначения и сравнивается с адресом из пакета IP_D. Если в какой-либо строке совпадение произошло, то адрес следующего маршрутизатора для данного пакета берется из данной строки.
2. *Вторая фаза* выполняется только в том случае, если во время первой фазы не произошло совпадения адресов. Она состоит в *поиске неспецифического маршрута*, общего для группы узлов, к которой относится и пакет с адресом IP_D. Для этого средствами IP заново просматривается таблица маршрутизации, причем с *каждой* записью производятся следующие действия:
   - маска (обозначим ее M), содержащаяся в данной записи, «накладывается» на IP-адрес узла назначения IP_D, извлеченный из пакета: IP_D AND M;
   - полученное в результате число сравнивается со значением, которое помещено в поле адреса назначения той же записи таблицы маршрутизации;
   - если происходит совпадение, то протокол IP соответствующим образом отмечает эту строку;
   - если просмотрены не все строки, то протокол IP аналогичным образом просматривает следующую строку, если все (включая строку о маршруте по умолчанию), то просмотр записей заканчивается и происходит переход к следующему шагу.

3. После просмотра всей таблицы маршрутизатор выполняет одно из трех действий:
   - если не произошло ни одного совпадения и маршрут по умолчанию отсутствует, то пакет отбрасывается;
   - если произошло одно совпадение, то пакет отправляется по маршруту, указанному в строке с совпавшим адресом;
   - если произошло несколько совпадений, то все помеченные строки сравниваются и выбирается маршрут из той строки, в которой количество совпавших двоичных разрядов наибольшее (другими словами, в ситуации, когда адрес назначения пакета принадлежит сразу нескольким подсетям, маршрутизатор использует наиболее специфический маршрут).

ПРИМЕЧАНИЕ

Во многих таблицах маршрутизации запись с адресом 0.0.0.0 и маской 0.0.0.0 соответствует маршруту по умолчанию. Действительно, любой адрес в пришедшем пакете после наложения на него маски 0.0.0.0 даст адрес сети 0.0.0.0, что совпадает с адресом, указанным в записи. Поскольку маска 0.0.0.0 имеет нулевую длину, этот маршрут считается самым неспецифическим и используется только при отсутствии совпадений с остальными записями из таблицы маршрутизации.

Обратимся к нашему примеру и посмотрим, как маршрутизатор R2 (см. рис. 14.10) использует описанный алгоритм для работы со своей таблицей маршрутизации (см. табл. 14.8). Пусть на маршрутизатор R2 поступает пакет с адресом назначения 129.44.78.200. Модуль IP, установленный на этом маршрутизаторе, прежде всего сравнит этот адрес с адресом 129.44.128.15, для которого определен специфический маршрут. В отсутствие совпадения модуль IP начинает последовательно обрабатывать все строки таблицы, накладывая маски и сравнивая результаты до тех пор, пока не найдет совпадения номера сети в адресе назначения и в строке таблицы. В результате определяется маршрут для пакета 129.44.78.200 — он должен быть отправлен на выходной порт маршрутизатора 129.44.64.7 в сеть 129.44.64.0, непосредственно подключенную к данному маршрутизатору.

Использование масок переменной длины

Во многих случаях более эффективным является разбиение сети на подсети разного размера. На рис. 14.11 приведен другой вариант распределения того же адресного пространства 129.44.0.0/16, что и в предыдущем примере. Здесь половина из имеющихся адресов (2^{15}) отведена для создания *сети 1*, имеющей адрес 129.44.0.0 и маску 255.255.128.0.

Следующая порция адресов, составляющая четверть всего доступного адресного пространства (2^{14}), определяемая адресом 129.44.128.0 с маской 255.255.192.0, назначена для *сети 2*. Затем в пространстве адресов был «вырезан» небольшой фрагмент для создания вспомогательной *сети 3*, предназначенной для связывания внутреннего маршрутизатора R2 с внешним маршрутизатором R1. Для нумерации узлов в такой вырожденной сети достаточно отвести два двоичных разряда. Из четырех возможных комбинаций номеров узлов — 00, 01, 10 и 11 — два номера имеют специальное назначение и не могут быть присвоены узлам, но оставшиеся два (10 и 01) позволяют адресовать порты маршрутизаторов. Поле номера узла в таком случае имеет два двоичных разряда, маска в десятичной нотации имеет вид 255.255.255.252, а номер сети, как видно из рисунка, равен 129.44.192.0.

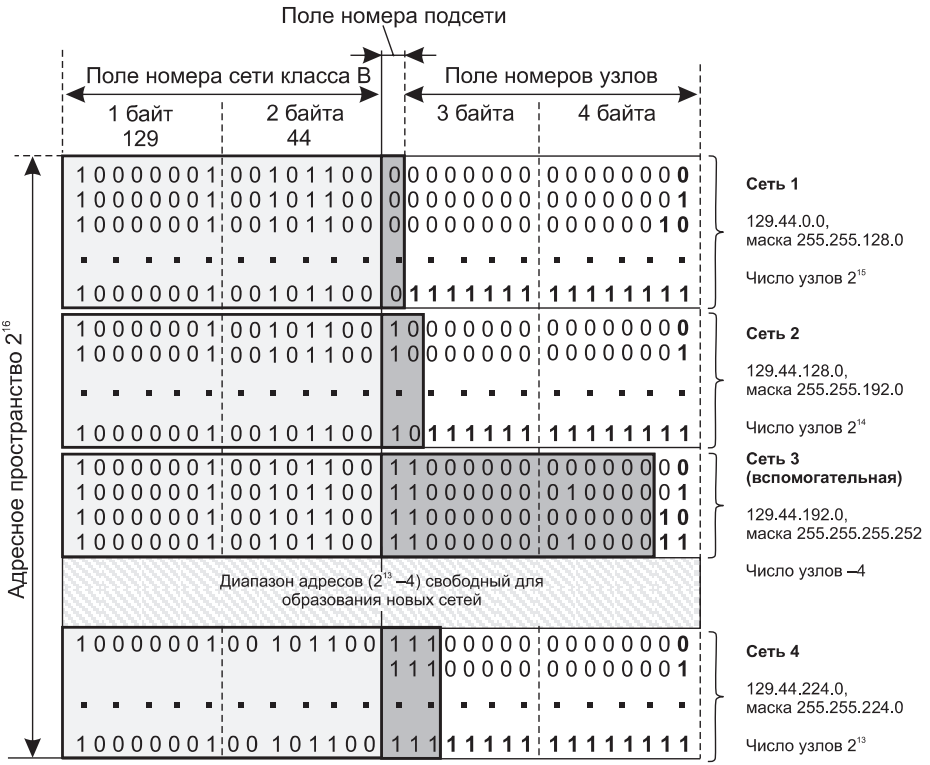


Рис. 14.11. Разделение адресного пространства 129.44.0.0 сети класса B на сети разного размера путем использования масок переменной длины

ПРИМЕЧАНИЕ

Глобальным связям между маршрутизаторами, соединенными по двухточечной схеме, не обязательно давать IP-адреса. Такой интерфейс маршрутизатора называется ненумерованным (unnumbered). Однако обычно подобной вырожденной сети все же дают IP-адрес. Помимо прочего, это делается, например, для того чтобы скрыть внутреннюю структуру сети и обращаться к ней по одному адресу входного порта маршрутизатора, в данном примере — по адресу 129.44.192.1, применяя технику трансляции сетевых адресов (Network Address Translation, NAT). Также на двухточечных линиях можно использовать адреса с маской /31, стандарт разрешает в этом случае использовать в поле номера узла значения 0 и 1.

Оставшееся адресное пространство администратор может «нарезать» на разное количество сетей разного объема, в зависимости от своих потребностей. Из оставшегося пула ($2^{14} - 4$) адресов администратор, например, может образовать еще одну достаточно большую сеть с числом узлов 2^{13} — на рисунке это *сеть 4*. Ясно, что разбиение может быть другим, но в любом случае с помощью масок переменного размера администратор имеет возможность более рационально распорядиться всеми имеющимися у него адресами. На рис. 14.12 показан пример сети, структура которой отражает описанное выше разделение адресного пространства.

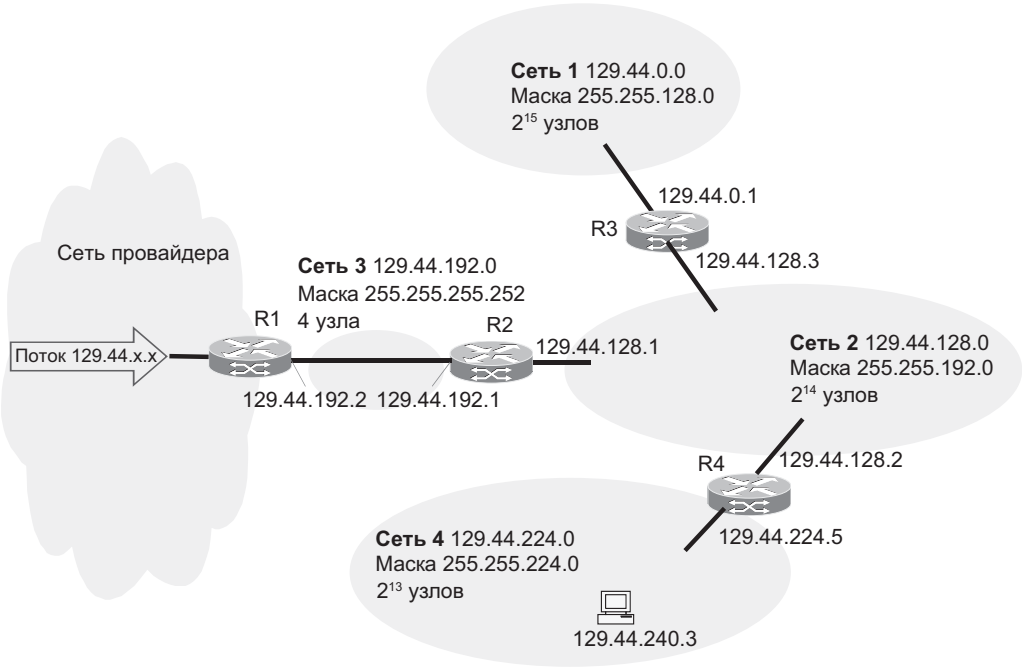


Рис. 14.12. Структуризация сети масками переменной длины

Давайте посмотрим, как маршрутизатор R2 обрабатывает поступающие на его интерфейсы пакеты (табл. 14.9).

Таблица 14.9. Таблица маршрутизатора R2 в сети с масками переменной длины

Адрес назначения	Маска	Адрес следующего маршрутизатора	Адрес порта	Расстояние
129.44.0.0	255.255.128.0	129.44.128.3	129.44.128.1	1
129.44.128.0	255.255.192.0	129.44.128.1	129.44.128.1	Подключена
129.44.192.0	255.255.255.248	129.44.192.1	129.44.192.1	Подключена
129.44.224.0	255.255.224.0	129.44.128.2	129.44.128.1	1
0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	—

Пусть поступивший на R2 пакет имеет адрес назначения 129.44.240.3. Поскольку специфические маршруты в таблице отсутствуют, маршрутизатор переходит ко второй фазе: для каждой строки таблицы маска накладывается на адрес пакета, а затем результат сравнивается с адресом назначения из таблицы маршрутизации:

(129.44.240.3) AND (255.255.128.0) = 129.44.128.0 — не совпадает с 129.44.0.0;
(129.44.240.3) AND (255.255.192.0) = 129.44.192.0 — не совпадает с 129.44.128.0;
(129.44.240.3) AND (255.255.255.248) = 129.44.240.0 — не совпадает с 129.44.192.0;
(129.44.240.3) AND (255.255.224.0) = 129.44.224.0 — совпадает.

Таким образом, совпадение имеет место в одной строке. Пакет будет отправлен на маршрутизатор R4, к которому подключена сеть назначения 129.44.224.0.

ПРИМЕЧАНИЕ

При использовании механизма маршрутизации на основе масок в IP-пакетах передается только IP-адрес назначения, маска сети назначения — нет. Поэтому из IP-адреса пришедшего пакета невозможно выяснить, какая часть адреса относится к номеру сети, а какая — к номеру узла. Если маски во всех подсетях имеют один размер, то это не создает проблем. Если же для образования подсетей применяют маски переменной длины, то маршрутизатор должен как-то узнавать, каким адресам сетей какие маски соответствуют. Для этого используются протоколы маршрутизации, переносящие между маршрутизаторами не только служебную информацию об адресах сетей, но и о масках, соответствующих этим номерам. К последним относятся протоколы RIPv2 и OSPF, тогда как протокол RIP маски не переносит и для маршрутизации на основе масок переменной длины не подходит.

(S) *Перекрытие адресных пространств*

CIDR и маршрутизация

Поскольку таблицы магистральных маршрутизаторов в интернете содержат сотни и даже тысячи маршрутов, то в сети могут возникать перегрузки, вызванные обработкой больших объемов служебной информации об обновлении таблиц. На решение этой проблемы направлена, в частности, технология бесклассовой междоменной маршрутизации CIDR. Мы уже говорили в предыдущей главе о том, как CIDR способствует гибкости распределения адресов между владельцами сетей. Теперь покажем, как эта технология повышает эффективность маршрутизации.

Суть заключается в следующем. Каждому поставщику услуг Интернета назначается *непрерывный* диапазон IP-адресов. При таком подходе *все* адреса каждого поставщика услуг имеют общую старшую часть — **префикс**, поэтому маршрутизация на магистралях Интернета может осуществляться на основе префиксов, а не полных адресов сетей. А это значит, что вместо множества записей по числу сетей в таблицу маршрутизации достаточно поместить *одну запись сразу для всех сетей*, *имеющих общий префикс*. Такое агрегирование адресов позволит уменьшить объем таблиц в маршрутизаторах всех уровней и, следовательно, ускорить работу маршрутизаторов и повысить пропускную способность Интернета.

Ранее рассматривались примеры, где администраторы сетей с помощью масок делили на несколько частей непрерывный пул адресов, полученный от поставщика услуг, чтобы использовать эти части для структуризации своей сети. Такой вариант применения масок для разделения на подсети называют **субнетингом (subnetting)**.

Вместе с тем в процессе разделения на подсети с помощью масок проявлялся и обратный эффект их применения — агрегирование. Упрощенно говоря, чтобы направить весь суммарный трафик, адресованный из внешнего окружения в сеть, разделенную на подсети, достаточно, чтобы во всех внешних маршрутизаторах наличествовала одна строка. В этой строке на месте адреса назначения должен быть указан *общий префикс для всех этих сетей*. Здесь мы имеем дело с операцией, обратной разделению на подсети, — операцией объединения нескольких меньших сетей в одну более крупную. Она называется **супернетингом (supernetting)**.

Вернемся к рис. 14.12 и зададимся вопросом, как могла бы выглядеть таблица маршрутизатора R1. Очевидный ответ — там должны быть записи обо всех четырех сетях клиента. Действительно, такой вариант будет работать. Но маршрутизация будет еще более эффективной, если в таблице маршрутизатора R1 задать только одну запись, которая выполняет *агрегирование* адресов всех подсетей, созданных на базе одной сети 129.44.0.0 (табл. 14.10). Вторая строка говорит о том, что среди всех возможных подсетей сети 129.44.0.0 есть одна (129.44.192.0/30), которой можно направлять пакеты непосредственно, а не через маршрутизатор R2.

Таблица 14.10. Фрагмент таблицы маршрутизатора R1

Адрес назначения	Маска	Адрес следующего маршрутизатора	Адрес порта	Расстояние
129.44.0.0	255.255.0.0	129.44.192.1	129.44.191.2	2
129.44.192.0	255.255.255.192	129.44.192.2	129.44.192.2	Подключена

Действуя аналогичным образом, провайдер отводит для каждого клиента по одной строке независимо от количества подсетей, организованных ими в своих сетях. Если все поставщики услуг интернета начнут придерживаться стратегии CIDR, то особенно заметный выигрыш будет достигаться в магистральных маршрутизаторах.

Необходимым условием эффективного использования технологии CIDR является **локализация адресов**, то есть назначение адресов, имеющих совпадающие префиксы, сетям, располагающимся *территориально по соседству*. Только в таком случае трафик может быть агрегирован.

К сожалению, сейчас распределение адресов носит во многом случайный характер. Кардинальный путь решения проблемы — перенумерование сетей. Однако эта процедура сопряжена с определенными временны́ми и материальными затратами, и для ее проведения пользователей нужно каким-либо образом стимулировать. В качестве таких стимулов рассматривается, например, введение оплаты за строку в таблице маршрутизации или же за количество узлов в сети. Первое требование подводит потребителя к мысли получить у поставщика услуг такой адрес, чтобы маршрутизация трафика в его сеть шла на основании префикса и номер его сети не фигурировал больше в магистральных маршрутизаторах. Требование оплаты каждого адреса узла также может подтолкнуть потребителя решиться на перенумерование, чтобы получить ровно столько адресов, сколько ему нужно.

Технология CIDR успешно используется в текущей версии протокола IP (IPv4) и поддерживается такими протоколами маршрутизации, как OSPF, RIP-2, BGP4 (в основном на магистральных маршрутизаторах интернета). Особенности применения CIDR в новой версии протокола IP (IPv6) рассматриваются в главе 17.

Фрагментация IP-пакетов

Важной особенностью протокола IP, отличающей его от других сетевых протоколов (например, от сетевого протокола IPX, который какое-то время назад конкурировал с IP), является его способность выполнять *динамическую фрагментацию* пакетов при передаче их между сетями с различными максимально допустимыми значениями длины поля данных кадров (Maximum Transmission Unit, **MTU**). Значения MTU зависят как от протокола, так и от настройки сетевых интерфейсов.

Прежде всего отметим разницу между фрагментацией сообщений *в узле-отправителе* и динамической фрагментацией сообщений *в транзитных узлах* сети — маршрутизаторах. В первом случае деление сообщения на несколько более мелких частей (фрагментация) происходит при передаче данных между протоколами стека в пределах одного и того же компьютера. Протоколы анализируют тип технологии нижнего уровня, определяют ее MTU и делят сообщения на части, умещающиеся в кадры канального уровня того же стека протоколов. В стеке TCP/IP эту задачу решает протокол TCP, который разбивает поток байтов, передаваемых ему с прикладного уровня, на сегменты нужного размера, например по 1460 байт, если на нижнем уровне данной сети работает протокол Ethernet. Протокол IP в узле-отправителе, как правило, не использует свои возможности по фрагментации пакетов. А вот на транзитном узле — маршрутизаторе, когда пакет необходимо передать из сети с большим значением MTU в сеть с меньшим значением MTU, способности протокола IP выполнять фрагментацию становятся востребованными. *Пакеты-фрагменты*, путешествуя по сети, могут вторично подвергнуться фрагментации на каком-либо из промежуточных маршрутизаторов.

Параметры фрагментации

Каждый из фрагментов снабжается полноценным заголовком IP. Для последующей *сборки* фрагментов в исходное сообщение используются следующие поля заголовка:

- **Идентификатор** пакета используется для *распознавания* пакетов, образовавшихся путем деления на части (фрагментации) исходного пакета. Все части (фрагменты) одного пакета должны иметь одинаковое значение этого поля. Модуль IP, отправляющий пакет, устанавливает в поле идентификатора значение, которое должно быть уникальным для данной пары отправителя и получателя в течение всего времени, пока данный пакет (или любой его фрагмент) может существовать в составной IP-сети.
- Поле **времени жизни TTL** (Time To Live) занимает один байт и определяет предельный срок, в течение которого пакет может перемещаться по сети. Время жизни пакета измеряется в секундах и задается источником (отправителем). Как отмечено в начале этой главы, по истечении каждой секунды пребывания на каждом из маршрутизаторов, через которые проходит пакет во время своего «путешествия» по сети, из его текущего времени жизни вычитается единица; единица вычитается и в том случае, если время пребывания было меньше секунды. Поскольку современные маршрутизаторы редко обрабатывают пакет дольше чем за одну секунду, время жизни можно интерпретировать как максимальное число транзитных узлов, которые разрешено пройти пакету. Если значение поля времени жизни становится нулевым до того, как пакет достигает получателя, то пакет уничтожается. При сборке фрагментов хост-получатель использует значение TTL как крайний срок ожидания недостающих фрагментов.
- Поле **смещения фрагмента** содержит информацию о положении фрагмента относительно начала поля данных исходного нефрагментированного пакета. Так, первый фрагмент будет иметь в поле смещения нулевое значение. В пакете, не разбитом на фрагменты, поле смещения также имеет нулевое значение. Смещение задается в байтах и должно быть кратно 8 байтам.
- Установленный в единицу однобитный флаг **MF** (More Fragments — больше фрагментов) говорит о том, что данный пакет является промежуточным (не последним) фрагментом. Модуль IP, отправляющий нефрагментированный пакет, устанавливает бит MF в ноль.

- Флаг **DF** (Do not Fragment — не фрагментировать), установленный в единицу, запрещает маршрутизатору фрагментировать данный пакет; если помеченный таким образом пакет не может достичь получателя без фрагментации, то модуль IP его уничтожает, а узлу-отправителю посылается диагностическое сообщение. Возможность запретить фрагментацию позволяет в некоторых случаях ускорить работу приложений. Для этого необходимо предварительно исследовать сеть, определить максимальный размер пакета, который сможет пройти весь путь без фрагментации, а затем использовать пакеты такого или меньшего размера.

Механизм фрагментации

Рассмотрим механизм фрагментации на примере сети, показанной на рис. 14.13.

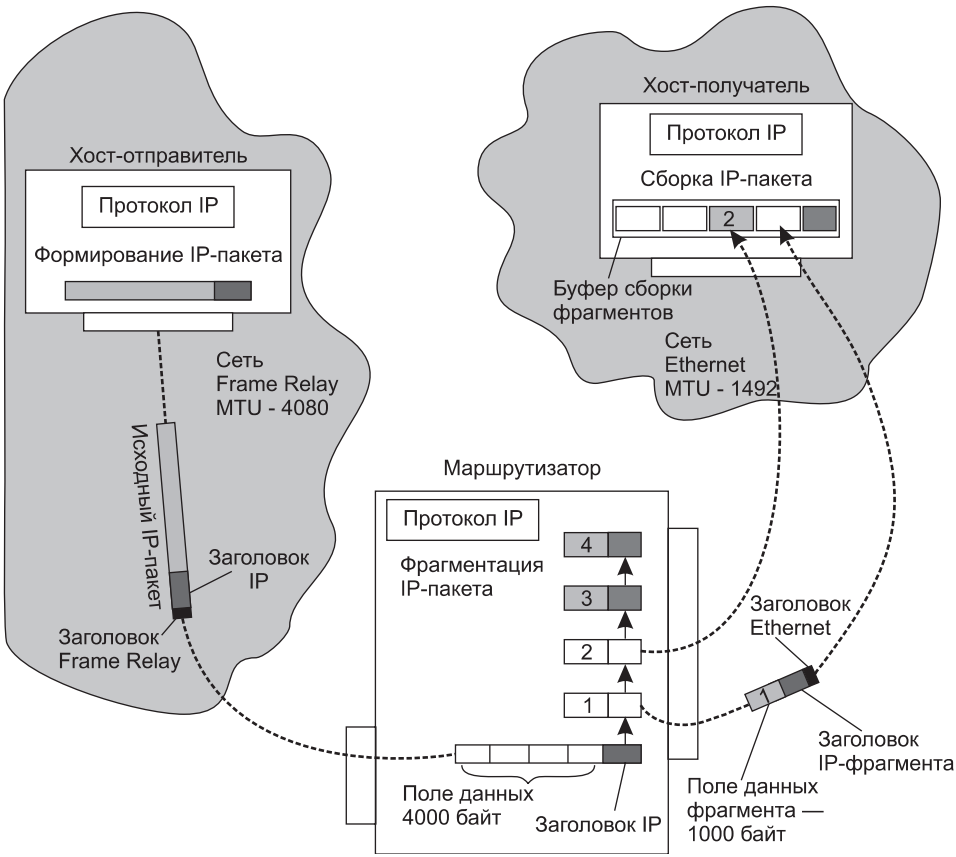


Рис. 14.13. Фрагментация в составной сети

В одной из подсетей (Frame Relay) значение MTU равно 4080, в другой (Ethernet) — 1492. Хост, принадлежащий сети Frame Relay, передает данные хосту в сети Ethernet. На обоих хостах и на маршрутизаторе, связывающем подсети, установлен стек протоколов TCP/IP.

Транспортному уровню *хоста-отправителя* известно значение MTU нижележащей технологии (4080). На основании этого модуль TCP «нарезает» свои сегменты размером 4000 байт и передает вниз протоколу IP, который помещает сегменты в поле данных IP-пакетов и генерирует для них заголовки. Обратим особое внимание на заполнение полей заголовка, прямо связанных с фрагментацией:

- пакету присваивается уникальный *идентификатор*, например 12456;
- поскольку пакет еще не фрагментирован, в поле *смещения* помещается значение 0;
- признак *MF* также обнуляется, это показывает, что пакет одновременно является и своим последним фрагментом;
- признак *DF* устанавливается в 1, это означает, что данный пакет можно фрагментировать.

Общая величина IP-пакета составляет 4000 плюс 20 (размер заголовка IP), то есть 4020 байт, что умещается в поле данных кадра Frame Relay, которое в данном примере равно 4080. Далее модуль IP хоста-отправителя передает этот кадр своему сетевому интерфейсу Frame Relay, который отправляет кадры следующему маршрутизатору. Модуль IP следующего маршрутизатора по сетевому адресу прибывшего IP-пакета определяет, что пакет нужно передать в сеть Ethernet. Однако она имеет значение MTU, равное 1492, что значительно меньше размера поступившего на входной интерфейс пакета. Следовательно, IP-пакет необходимо фрагментировать. Модуль IP выбирает размер поля данных фрагмента равным 1000, так что из одного большого IP-пакета получается четыре маленьких пакета-фрагмента. Для каждого фрагмента и его заголовка IP в маршрутизаторе создается отдельный буфер (на рисунке фрагменты и соответствующие им буферы пронумерованы от 1 до 4). Протокол IP копирует в эти буферы содержимое некоторых полей заголовка IP исходного пакета, создавая тем самым «заготовки» заголовков IP всех новых пакетов-фрагментов. Одни параметры заголовка IP копируются в заголовки всех фрагментов, другие — лишь в заголовок первого фрагмента.

В процессе фрагментации могут измениться значения некоторых полей заголовков IP в пакетах-фрагментах по сравнению с заголовком IP исходного пакета. Так, каждый фрагмент имеет собственные значения контрольной суммы заголовка, смещения фрагмента и общей длины пакета. Во всех пакетах, кроме последнего, флаг MF устанавливается в единицу, а в последнем фрагменте — в ноль. Полученные пакеты-фрагменты имеют длину 1020 байт (с учетом заголовка IP), поэтому они свободно помещаются в поле данных кадров Ethernet. На рисунке показаны разные стадии перемещения фрагментов по сети. Фрагмент 2 уже достиг хоста-получателя и помещен в приемный буфер. Фрагмент 1 еще перемещается по сети Ethernet, остальные фрагменты находятся в буферах маршрутизатора. Теперь обсудим, как происходит *сборка фрагментированного пакета на хосте назначения*.

ПРИМЕЧАНИЕ

Отметим, что IP-маршрутизаторы не собирают фрагменты пакетов в более крупные пакеты, даже если на пути встречается сеть, допускающая такое укрупнение. Это связано с тем, что отдельные фрагменты сообщения могут перемещаться по составной сети разными маршрутами, поэтому нет гарантии, что все фрагменты на своем пути пройдут через какой-то один определенный маршрутизатор.

На хосте назначения для каждого фрагментированного пакета отводится отдельный буфер. В этот буфер принимающий протокол IP помещает IP-фрагменты, у которых совпадают как IP-адреса отправителя и получателя, так и значения в полях идентифи-

катора (в нашем примере — 12456). Все эти признаки говорят модулю IP, что данные пакеты являются фрагментами одного исходного пакета. Сборка заключается в помещении данных из каждого фрагмента в позицию, определенную *смещением*, указанным в заголовке фрагмента.

Когда первый фрагмент исходного пакета приходит на хост-получатель, тот запускает *таймер*, определяющий максимальное время ожидания прибытия остальных фрагментов данного пакета. В различных реализациях IP применяются разные правила выбора максимального времени ожидания. В частности, таймер может быть установлен на фиксированный период времени (от 60 до 120 секунд), рекомендуемый RFC. Как правило, этот интервал достаточен для доставки пакета от отправителя получателю. В других реализациях максимальное время ожидания определяется с помощью адаптивных алгоритмов измерения и статистической обработки временны́х параметров сети, позволяющих оценивать ожидаемое время прибытия фрагментов. Наконец, тайм-аут может быть выбран и на основе значений TTL прибывающих фрагментов. Последний подход основан на том, что нет смысла ожидать, пока прибудут другие фрагменты пакета, если время жизни одного из прибывших фрагментов уже истекло. Если хотя бы один фрагмент пакета не успеет прийти на хост назначения к моменту истечения таймера, то никаких действий по дублированию отсутствующего фрагмента не предпринимается, а все полученные к этому времени фрагменты пакета отбрасываются. Хосту, пославшему исходный пакет, направляется ICMP-сообщение об ошибке. Такому поведению протокола IP вполне соответствует его кредо «по возможности» — стараться, но никаких гарантий не давать. Признаком окончания сборки является отсутствие незаполненных промежутков в поле данных и прибытие последнего фрагмента (с равным нулю флагом MF) до истечения тайм-аута. Когда пакет собран, протокол IP передает его вышележащему протоколу, например TCP.

Протокол ICMP

Протокол межсетевых управляющих сообщений (Internet Control Message Protocol, ICMP) является вспомогательным протоколом, использующимся для диагностики и мониторинга сети.

Можно представить ряд ситуаций, когда протокол IP не может доставить пакет адресату — например, истекает время жизни пакета, в таблице маршрутизации отсутствует маршрут к заданному в пакете адресу назначения, пакет не проходит проверку по контрольной сумме, шлюз не имеет достаточно места в своем буфере для передачи какого-либо пакета и т. д. и т. п. Свойство «необязательности» протокола IP, доставляющего данные «по возможности», компенсируется протоколами более высоких уровней стека TCP/IP, например TCP на транспортном уровне и в какой-то степени DNS на прикладном уровне. Они берут на себя обязанности по обеспечению надежности, применяя такие известные приемы, как нумерация сообщений, подтверждение доставки, повторная посылка данных.

Протокол ICMP также призван компенсировать ненадежность протокола IP, но несколько иным образом. Он не предназначен для *исправления* возникших при передаче пакета проблем: если пакет потерян, то ICMP не может послать его заново. Задача ICMP другая — он является *средством оповещения* отправителя о «несчастных случаях», произошедших с его пакетами. Пусть, например, протокол IP, работающий на каком-либо маршрутизаторе, обнаружил, что пакет для дальнейшей передачи по маршруту необходимо фрагментировать,

но в пакете установлен признак DF (не фрагментировать). В таком случае протокол IP, прежде чем отбросить пакет, отправляет *диагностическое* ICMP-сообщение конечному узлу-источнику. ICMP-сообщение передается по сети в поле данных IP-пакета. IP-адрес узла-источника определяется из заголовка исходного пакета, вызвавшего инцидент.

Сообщение, прибывшее в узел-источник, может быть обработано либо ядром ОС, либо протоколами транспортного и прикладного уровней, либо приложениями или проигнорировано. Важно, что обработка ICMP-сообщений не входит в обязанности протоколов IP и ICMP. Заметим, некоторые из пакетов могут исчезнуть в сети без каких-либо оповещений. Так, протокол ICMP не предусматривает передачу сообщений о проблемах, возникающих при обработке IP-пакетов, несущих ICMP-сообщения об ошибках. Такое решение принято разработчиками протокола, чтобы не порождать «штормы» в сетях, когда количество сообщений об ошибках лавинообразно возрастает.

Формат, типы и коды ICMP-сообщений

Особенностью протокола ICMP является функциональное разнообразие решаемых задач, а следовательно, и связанных с этим сообщений. Все типы сообщений имеют один и тот же формат (рис. 14.14), однако интерпретация полей существенно зависит от того, к какому типу относится сообщение.

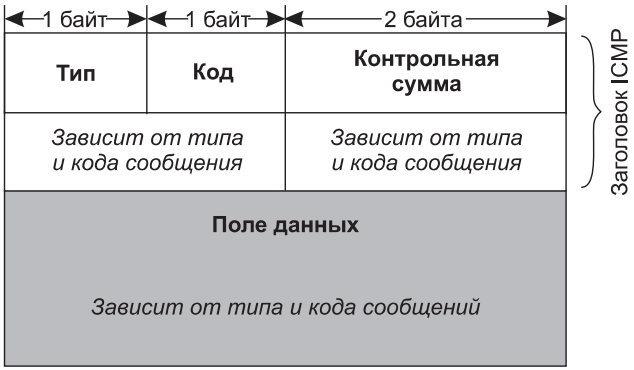


Рис. 14.14. Формат ICMP-сообщения

Заголовок ICMP-сообщения состоит из 8 байт:

- **тип** (1 байт) — числовой идентификатор типа сообщения;
- **код** (1 байт) — числовой идентификатор, более тонко дифференцирующий тип ошибки;
- **контрольная сумма** (2 байта) — подсчитывается для всего ICMP-сообщения.

Содержимое оставшихся 4 байт в заголовке и поле данных зависит от значений полей типа и кода.

На рис. 14.15 показана таблица основных типов ICMP-сообщений. Эти сообщения можно разделить на две группы:

- сообщения об *ошибках*;
- сообщения *запрос-ответ* (отмечены в таблице темным фоном).

Типы ICMP-сообщений

Значение в поле «Тип»	Тип сообщения
0	Эхо-ответ
3	Адресат недоступен
5	Перенаправить
8	Эхо-запрос
9	Объявление маршрутизатора
10	Запрос к маршрутизатору
11	Истечение времени
12	Проблема с параметрами пакета
13	Запрос отметки времени
14	Ответ отметки времени

Коды причин ошибки типа 3

Код	Причина
0	Сеть недостижима
1	Хост недостижим
2	Протокол недостижим
3	Порт недостижим
4	Требуется фрагментация
5	Ошибка в маршруте от источника
6	Сеть назначения неизвестна
7	Хост назначения неизвестен
...	

Рис. 14.15. Типы ICMP-сообщений и коды причин ошибки типа 3

Сообщения типа запрос-ответ связаны в пары: «эхо-запрос — эхо-ответ», «запрос маски — ответ маски», «запрос отметки времени — ответ отметки времени».

ICMP-сообщение (Type 9) «объявление маршрутизатора» (router advertisement) позволяет маршрутизаторам сообщать о своем присутствии в сети, а узлам — автоматически обнаруживать маршрутизаторы по умолчанию, а также получать другие полезные сетевые параметры. Этот тип сообщений часто используется в паре с ICMP-сообщением (Type 10) «Запрос маршрутизатора» (router solicitation), который узлы отправляют, когда хотят сами немедленно обнаружить маршрутизаторы, а не ждать их объявлений.

Маршрутизатор периодически (например, каждые несколько минут) отправляет объявление посредством групповой рассылки в локальной сети. Узлы получают объявление маршрутизатора и узнают его IP-адрес. Узлы добавляют маршрутизатор в качестве шлюза по умолчанию. Такая процедура является основой для автоконфигурации сетевых узлов. Например, когда ноутбук подключается к новой проводной или беспроводной сети, он отправляет сообщение «Запрос маршрутизатора», на которое маршрутизатор отвечает сообщением «Объявление маршрутизатора», после чего ноутбук автоматически устанавливает значение своего параметра «Маршрутизатор по умолчанию» на основе этого сообщения. Еще один пример использования сообщений «Объявление маршрутизатора» мы рассмотрим в следующем разделе «Мобильный IPv4».

Сообщения, относящиеся к группе сообщений об ошибках, конкретизируются уточняющим кодом. На рисунке показан фрагмент таблицы кодов сообщений об ошибке недостижимости узла назначения, имеющей тип 3. Как видим, сообщение этого типа может быть вызвано различными причинами: неверный адрес сети или конечного узла (код 0 или 1), отсутствие на конечном узле-адресате необходимого протокола прикладного уровня (код 2 — «протокол недостижим») или открытого порта UDP/TCP (код 3 — «порт недостижим»). Узел

(или сеть) назначения может быть также недостижим по причине временной неработоспособности аппаратуры или из-за того, что маршрутизатор не имеет данных о пути к сети назначения. Всего таблица содержит 15 кодов. Аналогичные таблицы кодов существуют и для других типов сообщений об ошибках.

Ошибка недостижимости узла и утилита traceroute

В качестве примера рассмотрим использование ICMP-сообщений об ошибке недостижимости узла в популярной утилите **traceroute**, предназначенной для мониторинга сети. Когда маршрутизатор не может передать или доставить IP-пакет, он отсылает узлу, отправившему этот пакет, сообщение о недостижимости узла назначения. Формат этого сообщения показан на рис. 14.16. В поле типа помещается значение 3, а в поле кода — значение из диапазона 0–15, уточняющее причину, по которой пакет не был доставлен. Следующие за полем контрольной суммы четыре байта заголовка *не используются* и заполняются нулями.

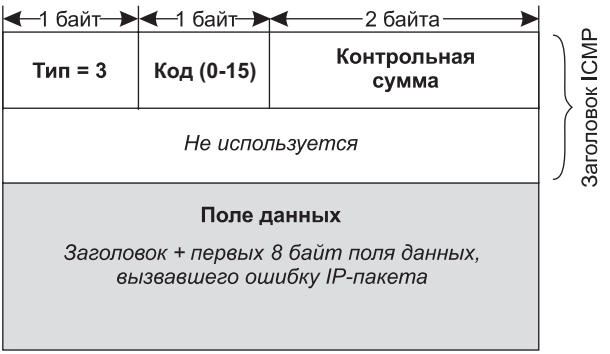


Рис. 14.16. Формат ICMP-сообщения об ошибке недостижимости узла назначения

Помимо причины ошибки, указанной в заголовке (в полях типа и кода), дополнительная диагностическая информация передается в поле данных ICMP-сообщения. Именно туда помещаются *заголовок IP и первые 8 байт данных* IP-пакета, вызвавшего ошибку. Эта информация позволяет узлу-отправителю еще точнее диагностировать причину ошибки, поскольку все протоколы стека TCP/IP, использующие для передачи сообщений IP-пакеты, помещают наиболее важную для анализа информацию в первые 8 байт своих сообщений. Так, в поле данных ICMP-сообщения могут оказаться первые 8 байт заголовка TCP или UDP, содержащих информацию (номер порта), идентифицирующую приложение, послывшее потерянный пакет. Значит, при разработке приложения можно предусмотреть встроенные средства реакции на сообщения о недоставленных пакетах. ICMP-сообщения об ошибках лежат в основе работы популярной утилиты traceroute для ОС Unix, имеющей в ОС Windows название **tracert**. Утилита позволяет проследить маршрут до удаленного хоста, определить среднее время оборота (RTT), IP-адрес и в некоторых случаях доменное имя каждого промежуточного маршрутизатора. Эта информация помогает найти маршрутизатор, на котором оборвался путь пакета к удаленному хосту.

Утилита traceroute осуществляет трассировку маршрута, посылая серию обычных IP-пакетов в конечную точку изучаемого маршрута. Идея метода состоит в следующем.

Значение времени жизни (TTL) первого отправляемого пакета устанавливается равным 1. Когда протокол IP первого маршрутизатора принимает этот пакет, он в соответствии со своим алгоритмом уменьшает значение TTL на 1 и получает 0. Маршрутизатор отбрасывает пакет с нулевым временем жизни и возвращает узлу-источнику ICMP-сообщение об ошибке истечения времени дейтаграммы (значение поля типа равно 11) вместе с заголовком IP и первыми 8 байтами потерянного пакета. Получив ICMP-сообщение о причине недоставки пакета, утилита traceroute запоминает адрес первого маршрутизатора (который извлекает из заголовка IP-пакета, несущего ICMP-сообщение).

Затем traceroute посылает следующий IP-пакет, но теперь со значением TTL, равным 2. Этот пакет благополучно проходит первый маршрутизатор, но «умирает» на втором, о чем немедленно отправляется аналогичное ICMP-сообщение об ошибке истечения времени дейтаграммы. Утилита traceroute запоминает адрес второго маршрутизатора, и т. д. Такие действия выполняются с каждым маршрутизатором вдоль маршрута вплоть до узла назначения или неисправного маршрутизатора. Мы рассматриваем работу утилиты traceroute весьма схематично, но и этого достаточно, чтобы оценить изящество идеи, лежащей в основе ее работы. Остальные ICMP-сообщения об ошибках имеют такой же формат и отличаются друг от друга только значениями полей типа и кода. Далее приведена копия экранной формы, выведенной утилитой tracert (ОС Windows) при трассировке хоста ds.internic.net [198.49.45.29]:

```
1 311 ms 290 ms 261 ms 144.206.192.100
2 281 ms 300 ms 271 ms 194.85.73.5
3 2023 ms 290 ms 311 ms moscow-m9-2-S5.relcom.eu.net [193.124.254.37]
4 290 ms 261 ms 280 ms MSK-M9-13.Relcom.EU.net [193.125.15.13]
5 270 ms 281 ms 290 ms MSK.RAIL-1-ATM0-155Mb.Relcom.EU.net [193.124.254.82]
6 300 ms 311 ms 290 ms SPB-RASCOM-1-E3-1-34Mb.Relcom.EU.net [193.124.254.78]
7 311 ms 300 ms 300 ms Hssi11-0.GW1.STK2.ALTER.NET [146.188.33.125]
8 311 ms 330 ms 291 ms 421.ATM6-0-0.CR2.STK2.Alter.Net [146.188.5.73]
9 360 ms 331 ms 330 ms 219.Hssi4-0.CR2.LND1.Alter.Net [146.188.2.213]
10 351 ms 330 ms 331 ms 412.Atm5-0.BR1.LND1.Alter.net [146.188.3.205]
11 420 ms 461 ms 420 ms 167.ATM8-0-0.CR1.ATL1.Alter.Net [137.39.69.182]12 461 ms 441
ms 440 ms 311.ATM12-0-0.BR1.ATL1.Alter.Net [137.39.21.73]13 451 ms 410 ms 431 ms
atlanta1-br1.bbnplanet.net [4.0.2.141]14 420 ms 411 ms 410 ms vienna1-br2.bbnplanet.
net [4.0.3.154]15 411 ms 430 ms 2514 ms vienna1-nbr3.bbnplanet.net [4.0.3.150]16
430 ms 421 ms 441 ms vienna1-nbr2.bbnplanet.net [4.0.5.45]17 431 ms 451 ms 420
ms cambridge1-br1.bbnplanet.net [4.0.5.42]18 450 ms 461 ms 441 мс cambridge1-
cr14.bbnplanet.net [4.0.3.94]19 451 мс 461 мс 460 мс attbcstoll.bbnplanet.net
[206.34.99.38]20 501 мс 460 мс 481 мс shutdown.ds.internic.net [198.49.45.29]
```

Последовательность строк соответствует последовательности маршрутизаторов, образующих маршрут к заданному узлу. Первое число в строке — *число хопов* до соответствующего маршрутизатора. Утилита traceroute тестирует каждый маршрутизатор трижды, поэтому следующие три числа в строке — это значения *RTT*, вычисленные путем посылки трех пакетов, время жизни которых истекло на этом маршрутизаторе. Если ответ от какого-либо маршрутизатора не приходит за заданное время, то вместо времени на экране печатается звездочка (*). Далее идут *доменное имя* (если оно имеется) и *IP-адрес* маршрутизатора. Видно, что почти все интерфейсы маршрутизаторов поставщиков услуг интернета зарегистрированы в службе DNS, а первые два, относящиеся к локальным маршрутизаторам, — нет.

Еще раз подчеркнем, что время, указанное в каждой строке, не отражает время прохождения пакетов между двумя соседними маршрутизаторами — это время, за которое пакет проделывает путь от источника до соответствующего маршрутизатора и обратно. Так как ситуация в интернете с загрузкой маршрутизаторов постоянно меняется, время достижимости маршрутизаторов не всегда нарастает монотонно, а может изменяться достаточно произвольным образом.

Сообщения эхо-запрос и эхо-ответ в утилите ping

А сейчас давайте рассмотрим представителей другой группы ICMP-сообщений — запрос-ответ — и поговорим об использовании этих сообщений в утилите **ping**. Эхо-запрос и эхо-ответ, в совокупности называемые **эхо-протоколом**, представляют собой очень простое, но эффективное средство мониторинга сети. Компьютер или маршрутизатор посылает по составной сети ICMP-сообщение эхо-запроса, указывая в нем IP-адрес узла, достижимость которого нужно проверить. Узел, получивший эхо-запрос, формирует и отправляет эхо-ответ отправителю запроса. Так как эхо-запрос и эхо-ответ передаются по сети внутри IP-пакетов, их успешная доставка указывает на нормальное функционирование всей транспортной системы составной сети.

Формат эхо-запроса и эхо-ответа показан на рис. 14.17. Поле типа для эхо-ответа равно 0, для эхо-запроса — 8; поле кода всегда равно 0 и для запроса, и для ответа. В байтах 5 и 6 заголовка содержится *идентификатор запроса*, в байтах 7 и 8 — *порядковый номер*. В поле данных эхо-запроса может быть помещена произвольная информация, которая в соответствии с данным протоколом должна быть скопирована в поле данных эхо-ответа.



Рис. 14.17. Формат ICMP-сообщений типа эхо-запрос и эхо-ответ

Поля идентификатора запроса и порядкового номера *используются одинаковым образом всеми сообщениями типа запрос-ответ*. Посылая запрос, приложение помещает в эти два поля информацию, предназначенную для последующего встраивания ее в соответствующий ответ. Сообщение-ответ копирует значения этих полей в свои поля того же назначения. Когда ответ возвращается в пункт отправки сообщения-запроса, то на основании идентификатора он может найти и опознать приложение, пославшее запрос. А порядковый номер используется приложением, чтобы связать полученный ответ с соответствующим запросом (учитывая, что одно приложение может выдать несколько однотипных запросов).

Утилита ping обычно посылает серию эхо-запросов к тестируемому узлу и предоставляет пользователю статистику об утерянных эхо-ответах и среднем времени реакции сети на запросы. Утилита ping выводит на экран сообщения следующего вида обо всех поступивших ответах:

```
# ping server1.citmgu.ru
Pinging server1.citmgu.ru [193.107.2.200] with 64 bytes of data:
Reply from 193.107.2.200: bytes=64 time=256ms TTL= 123
Reply from 193.107.2.200: bytes=64 time=310ms TTL= 123
Reply from 193.107.2.200: bytes=64 time=260ms TTL= 123
Reply from 193.107.2.200: bytes=64 time=146ms TTL= 123
```

Из сообщения видно, что в ответ на тестирующие запросы, посланные узлу **server1.mgu.ru**, получено 4 эхо-ответа. Длина каждого сообщения — 64 байта. В следующей колонке помещены значения времени оборота (RTT), то есть времени от момента отправки запроса до получения ответа на запрос. Как видим, сеть работает достаточно нестабильно — время в последней строке отличается от времени во второй более чем в два раза. На экран выводится также оставшееся время жизни поступивших пакетов.

Мобильный IPv4

Мобильный IP (Mobile IP, MIP, RFC 5944) изначально был разработан IETF для использования в проводных сетях IPv4, между которыми пользователи перемещаются со своими ноутбуками. Позже была разработана версия **Mobile IPv6** (RFC 6275), которая также ориентировалась на проводные сети, но теперь уже сети IPv6. И наконец, новая версия прокси мобильного IPv6 (**Proxy MobileIPv6**, RFC 5213) была предназначена для использования в беспроводных сетях — сетях Wi-Fi (при перемещении между точками доступа, принадлежащими различным IP-сетям) и мобильных сотовых сетях.

> Проблема мобильности для всех упомянутых протоколов формулировалась одинаково: как сделать так, чтобы узел, переместившийся из одной IP-сети в другую IP-сеть, где ему был присвоен новый адрес, мог продолжать коммуникации, в которых он участвовал под старым адресом.

Всем знакомы почтовые проблемы, вызванные переездом. Когда мы меняем адрес (переезжаем в другую «сеть»), надо позаботится о том, чтобы письма, направленные нам по старому адресу, продолжали к нам поступать. У нас есть несколько вариантов:

- Пойти в почтовый офис и зарегистрировать там заявку на получение услуги переадресации почтовых отправлений.
- Мы можем также сообщить новый адрес всем адресатам, от которых хотим продолжать получать письма.
- Можно оставить свой новый адрес консьержу или другу и попросить их регулярно просматривать наш почтовый ящик и переправлять поступившие письма, то есть упаковывать письма в другие конверты с написанным на них новым адресом.

Каждый из указанных подходов в том или ином виде может быть реализован для перемещающихся IP-узлов. Последняя аналогия (с консьержем) имеет наиболее близкое сходство с идеей обеспечения мобильности технологией Mobile IPv4.



Рис. 14.18. Схема работы Mobile IP

На рис. 14.18 показаны две связанные между собой IP-сети. Одна из них является *домашней* для узла, имеющего адрес IP1. Пусть этот узел выполняет клиентскую часть распределенного приложения, поддерживая сессию с сервером. Предположим, что ноутбук переместился в *гостевую* сеть, где ему был назначен новый адрес — IP2. Тогда сообщения от сервера, которые тот продолжает посылать на старый адрес IP1, не доходят до адресата. В результате нарушаются связь между частями приложения, а также другие коммуникации мобильного узла, в которых он участвовал под адресом IP1.

Именно эту проблему решает протокол Mobile IP. На мобильном устройстве и на маршрутизаторах домашней и гостевой сетей должны быть установлены и активированы дополнительные программные модули — программные агенты, выполняющие дополнительные функции по обеспечению мобильности. Агент в домашней сети называется **домашним агентом** (Home Agent), а агент во внешней сети называется **гостевым агентом** (Foreign Agent).

Домашний агент устанавливается на выделенном для этой функции маршрутизаторе домашней сети мобильного узла. Выбор маршрутизатора осуществляется с учетом путей прохождения трафика через сеть так, чтобы весь трафик к мобильному узлу проходил через этот маршрутизатор. Как правило, им является маршрутизатор, соединяющий тупиковую сеть, в которой находится мобильный узел, с интернетом или магистралью корпоративной сети.

Чтобы домашний агент знал, что ему предстоит отслеживать перемещения некоторого мобильного устройства и перенаправлять пакеты, посланные этому устройству по его IP-адресу в домашней сети, мобильное устройство должно *зарегистрироваться* у домашнего агента. Домашний агент периодически рассылает ICMP-сообщения «Объявление маршрутизатора» (Router Advertisement), и мобильное устройство, получив это сообщение, отвечает домашнему агенту запросом на регистрацию, во время которой оно сообщает агенту свой домашний адрес IP1, имеющий стандартное название — **мобильный адрес (MA).**

При перемещении в гостевую сеть мобильное устройство должно пройти процедуру регистрации и в ней. Эта процедура похожа на процедуру регистрации у домашнего агента и также использует ICMP-сообщения «Объявление маршрутизатора», которые в данном случае периодически распространяет гостевой агент. Последний указывает в ICMP-сообщении IP-адрес из диапазона адресов гостевой сети — **адрес COA** (Care Of Address), который предлагается вновь прибывшему узлу. На английском языке выражение Care Of Address используется для пометок на конверте «для такого-то», указывающих, что на самом деле письмо должно быть передано не адресату, указанному в поле адреса, а тому, чье имя указано в данной пометке. В нашем примере COA-адрес — это новый адрес мобильного устройства IP2. Приняв сообщение, мобильное устройство должно подтвердить намерение присоединиться к гостевой сети, отправив запрос на регистрацию гостевому агенту. В этом запросе указываются несколько параметров:

- IP-адрес домашнего агента;
- COA-адрес (IP2);
- адрес мобильного устройства в домашней сети MA (в примере — IP1) и
- срок действия регистрации.

Если гостевой агент согласен принять запрос на регистрацию (он проверяет аутентичность мобильного пользователя с помощью домашнего агента пользователя или же с помощью централизованной службы аутентификации, если она реализована в домашней сети), то он запоминает адреса COA и MA и пересылает запрос мобильного устройства домашнему агенту, используя его адрес. Домашний агент, приняв это сообщение, также проверяет аутентичность пользователя и, если запрос аутентичен, тоже запоминает адреса MA и COA. В завершение домашний агент посылает подтверждение гостевому агенту.

После этого между мобильным узлом и домашним агентом *устанавливается туннель*, то есть весь трафик, адресованный мобильному устройству, перехватывается маршрутизатором с установленным на нем домашним агентом, в котором этот мобильный узел зарегистрировался, каждый из поступающих пакетов инкапсулируется в новый IP-пакет с *новым* адресом назначения COA (IP2) и отправляется в сеть.

Перечислим основные этапы обмена пакетами между стационарным и мобильным устройствами после образования туннеля между домашним агентом и мобильным узлом.

- Пусть некоторое стационарное устройство посылает пакет мобильному устройству по его домашнему адресу (IP1).
- Когда пакет достигает домашней подсети, для этого адреса выполняется ARP-запрос. MAC-адрес, полученный в ответ на ARP-запрос, указывает на маршрутизатор, на котором установлен домашний агент.
- Домашний агент получает этот пакет, и вместо стандартной процедуры обработки IP-пакетов маршрутизатором, упаковывает его в другой IP-пакет с новым адресом назначения COA (IP2) и отправляет его по туннелю мобильному устройству.
- Мобильный узел получает и обрабатывает инкапсулированный пакет, извлекая оригинальный IP-пакет.
- Ответные пакеты от мобильного устройства к стационарному устройству отправляются без использования туннеля, так как адрес стационарного устройства маршрутизаторы будут обрабатывать стандартным образом, то есть обратный трафик «не обязан» проходить через маршрутизатор с установленным на нем домашним агентом.

Запись о регистрации устройства мобильного пользователя в гостевой сети удаляется из списка записей домашнего агента в одном из двух случаев — либо при регистрации устройства в домашней сети по его возвращении, либо по истечении срока действия регистрации.

Мы рассмотрели работу технологии Mobile IP для версии протокола IPv**4**, описание этой технологии для IPv6 вы можете найти в материалах на сайте книги. Кроме того, вопросы мобильности рассматриваются в части VI в разделах, посвященных общим принципам работы беспроводных сетей, мобильным сотовым сетям и сетям дронов.

Несмотря на то что в теории технология Mobile IP решает проблему мобильности, на практике она применяется редко. Причин здесь несколько, среди них:

- сложность в реализации: необходимо установить и сконфигурировать на узлах сети дополнительное программное обеспечение домашних и гостевых агентов;
- трансляция IP-адресов с помощью техники NAT (она рассматривается в главе 32) затрудняет использование Mobile IP;
- недостаточная производительность домашнего агента может привести к появлению узкого места в сети, кроме того, домашний агент является дополнительной целью атак на безопасность сети.

(S) *Технологии MobileIPv6 и Proxy MobileIPv6*

Поддержка QoS протоколом IP

Технологии стека TCP/IP были разработаны для эластичного трафика, который достаточно терпим к задержкам и вариациям задержек пакетов. Поэтому основное внимание разработчиков протоколов TCP/IP было сосредоточено на обеспечении надежной передачи трафика с помощью TCP. Однако со временем интернет стал передавать трафик и других типов, например потоки интерактивного голосового и видео трафика, которые были чувствительны к задержкам и их вариациям, и для качественной передачи требовали применения механизмов QoS (описанных в главе 5). Эти механизмы стали реализовываться производителями IP-маршрутизаторов, но какое-то время эти механизмы использовались каждым сетевым администратором по своему усмотрению, без какой-либо стройной системы. И только в середине 90-х годов начались работы по созданию стандартов QoS для IP-сетей, на основе которых можно было бы создать систему поддержки параметров QoS в масштабах составной сети и даже интернета.

В результате были разработаны две системы стандартов QoS для IP-сетей:

- система **интегрированного обслуживания (Integrated Services, IntServ)** ориентирована на предоставление гарантий QoS для потоков *конечных пользователей* «из конца в конец»;
- система **дифференцированного обслуживания (Differentiated Services, DiffServ)** предоставляет гарантии QoS в агрегированной форме *для классов трафика*.

Технология IntServ основана на сигнальном протоколе RSVP, который перед передачей потока трафика с требуемым качеством обслуживания должен пройти через все IP-маршрутизаторы на пути трафика от узла-источника до узла назначения и зарезервировать необходимые ресурсы (часть пропускной способности интерфейсов, ресурсы процессора

и т. п.) на каждом маршрутизаторе. Технология IntServ не нашла применения в интернете по нескольким причинам, среди которых основными являются две:

- Низкая масштабируемость, так как для каждого потока, а на магистралях интернета их могут быть десятки тысяч, маршрутизатор должен хранить информацию о резервировании ресурсов.
- Сложность кооперации между всеми провайдерами интернета, так как услуга предоставляется «из конца в конец» и для ее реализации необходима согласованная работа всех провайдеров на пути трафика. Такая кооперация требует сложных взаиморасчетов между провайдерами и наличия доверия, так как необходимо допустить в свою сеть чужой запрос для резервирования своих ресурсов для чужих пользователей.

Дифференцированное обслуживание (DiffServ) определяет поведение *отдельного* маршрутизатора по поддержанию необходимого качества обслуживания, не требуя согласованной работы маршрутизаторов на пути от узла-источника до узла-назначения. Такое поведение маршрутизаторов в отношении поддержки QoS названо в стандарте DiffServ «пошаговым» (per-hop behaviour). Оно не гарантирует качества обслуживания «из конца в конец», а предоставляет услуги QoS «по возможности».

Еще одним упрощением технологии DiffServ по сравнению с IntServ является работа с агрегированными **классами трафика**, а не с отдельными потоками. Пакеты IP агрегируются в класс в соответствии с требованиями QoS данных, которые они переносят. Например, один класс для пакетов, переносящих голосовой трафик, требующий гарантированного низкого уровня задержек, другой класс — для пакетов, переносящих эластичный трафик, устойчивый к задержкам, но чувствительный к потерям пакетов, и т. д. В отличие от потока, в классах трафика пакеты не различаются по их маршрутам. Именно такое объединение в класс и позволяет маршрутизатору обрабатывать их пошагово, без учета предыстории их прохождения через сеть.

Предполагается, что маршрутизатор использует имеющиеся у него в распоряжении механизмы QoS для требуемого обслуживания каждого класса — приоритетные и взвешенные очереди, профилирование и сглаживание трафика.

В качестве признака принадлежности IP-пакета к определенному классу в DiffServ используется значение метки, переносимой в поле приоритета IP-пакета (байт ToS), которое с появлением стандартов DiffServ было переопределено и названо байтом DS. Эта метка должна переносить между маршрутизаторами информацию о принадлежности пакета к определенному классу. Предполагается, что значение метки DS определяется либо приложением узла-отправителя, которое знает требования своего трафика и передает значение DS на уровень протокола IP, либо пограничным маршрутизатором на входе в сеть по правилам политики обработки входного трафика заданными администратором сети.

Как показано на рис. 14.19, байт DS переопределяет значения битов байта ToS, определенных ранее в соответствующих спецификациях (RFC 791, RFC 1122, RFC 1349).

В настоящее время используются только старшие шесть битов байта DS, причем из них только старшие три требуются для определения класса трафика (что дает не более восьми различных классов). Младший бит (из используемых шести) байта DS обычно переносит признак IN — индикатор того, что пакет вышел из профиля трафика и его можно отбросить или перевести в другой класс. Промежуточные два бита обычно описывают различные варианты обслуживания пакетов внутри одного класса трафика.

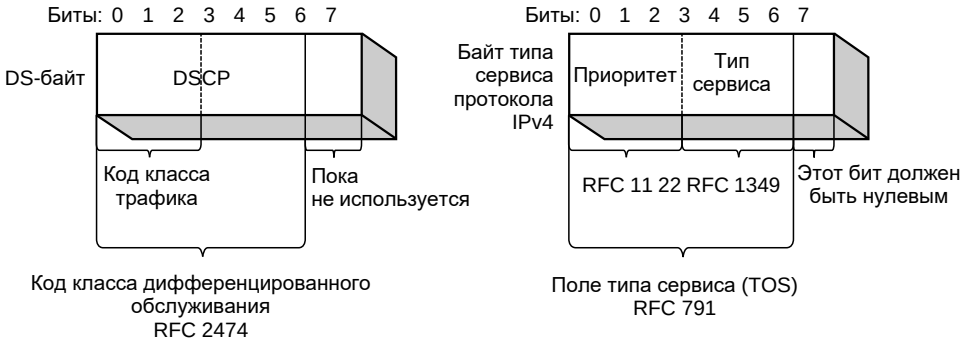


Рис. 14.19. Соответствие битов байта DS битам поля типа сервиса

На сегодняшний день в IETF разработано три стандарта пошагового продвижения пакетов для схемы PHB, которые представляют два разных варианта обслуживания.

- Быстрое продвижение (Expedited Forwarding, EF) характеризуется значением кода 10111 в байте DS (десятичное значение 46) и представляет собой высший уровень качества обслуживания, обеспечивая минимум задержек и вариаций задержек.
- Гарантированная доставка (Assured Forwarding, AF) характеризуется четырьмя классами трафика и тремя уровнями отбрасывания пакетов в каждом классе — всего получается 12 различных вариантов трафика с кодами DS от 10 до 38 (только четные значения). Каждому классу трафика выделяется определенный минимум пропускной способности и размер буфера для хранения его очереди. Трафик, параметры которого превышают указанные в профиле, доставляется с меньшей степенью вероятности, чем трафик, удовлетворяющий условиям профиля. Это означает, что качество его обслуживания может быть понижено, но он не обязательно будет отброшен.
- Доставка с усилиями ниже, чем «по возможности» (Lower-Effort, LE), — трафик этого класса должен обслуживаться в последнюю очередь, после трафика классов EF и AF, и даже после обычного трафика «по возможности» (Best Effort, BE), которым является трафик с нулевым значением поля DS, то есть непомеченного трафика. Класс LE рассчитан на передачу больших объемов данных, которые допускают передачу в течение длительного времени, например трафика сохранения резервных копий больших баз данных.

Основное назначение EF-сервиса — обеспечение высшего качества обслуживания, сопоставимого с качеством обслуживания выделенных каналов. Поскольку EF-сервис допускает полное вытеснение другого трафика (например, при реализации EF-сервиса с помощью приоритетной очереди), то его реализация должна включать некоторые средства ограничения влияния EF-трафика на другие классы трафика, например, путем ограничения скорости EF-трафика на входе маршрутизатора по алгоритму ведра маркеров. Максимальная скорость EF-трафика и, возможно, величина пульсаций должны устанавливаться сетевым администратором.

Четыре класса AF-сервиса ориентированы на гарантированную доставку, но без минимизации уровня задержек пакетов, как это оговорено для EF-сервиса. Гарантированная доставка выполняется в том случае, когда входная скорость трафика не превышает отведенной данному классу минимальной пропускной способности. Реализация классов AF-трафика хо-

рошо сочетается с EF-сервисом: EF-трафик может обслуживаться по приоритетной схеме, но с ограничением интенсивности входного потока. Оставшаяся пропускная способность распределяется между классами AF-трафика в соответствии с алгоритмом взвешенного обслуживания, который обеспечивает необходимую пропускную способность, но не минимизацию задержек. Реализация AF-сервиса предполагает (но не требует) взвешенного обслуживания для каждого класса с резервированной полосой пропускания, а также применения обратной связи (в форме RED).

Относительная простота определяет недостатки дифференцированного обслуживания. Главным из них является сложность предоставления количественных гарантий качества обслуживания «из конца в конец» в масштабах интернета. Поэтому технология DiffServ обычно реализуется в пределах одного административного домена — сети определенного провайдера услуг интернета, корпоративной сети или же домашней сети.

(S) *Протоколы IntServ и DiffServ. Поддержка QoS маршрутизаторами*

ГЛАВА 15 Протоколы транспортного уровня TCP и UDP

Мультиплексирование и демультиплексирование приложений

Порты

Каждый компьютер способен выполнять несколько процессов, более того, даже отдельный прикладной процесс может иметь несколько точек входа, выступающих в качестве адресов назначения для пакетов данных. Поэтому доставка данных на сетевой интерфейс компьютера-получателя — это еще не конец пути, так как данные необходимо переправить конкретному процессу-получателю. Реализуемая протоколами TCP и UDP процедура распределения между прикладными процессами пакетов, поступающих от сетевого уровня, называется **демультиплексированием** (рис. 15.1).

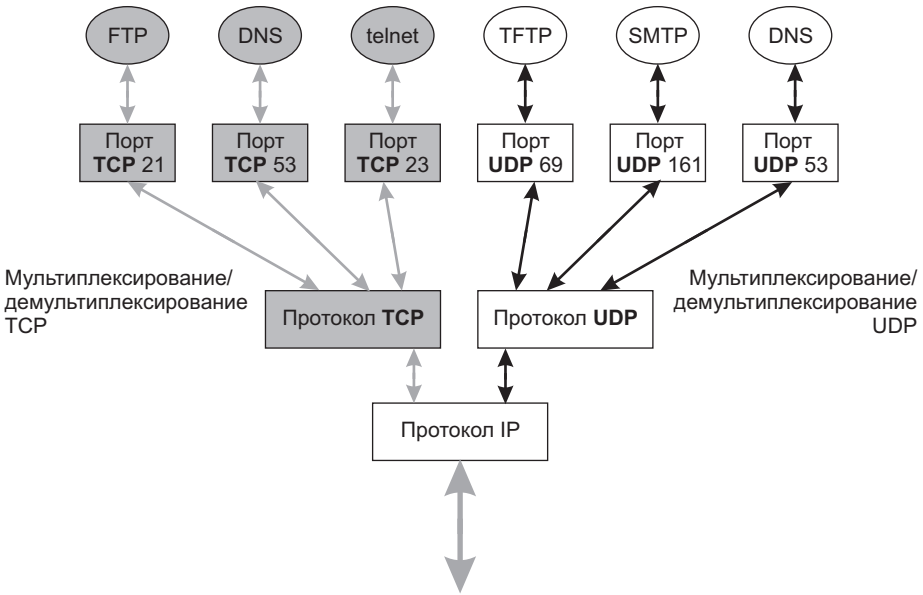


Рис. 15.1. Мультиплексирование и демультиплексирование на транспортном уровне

Существует и обратная задача: данные, генерируемые разными приложениями, работающими на одном конечном узле, должны быть переданы общему для всех них протокольному модулю IP для последующей отправки в сеть. Эту работу, называемую **мультиплексированием**, тоже выполняют протоколы TCP и UDP.

Протоколы TCP и UDP ведут для каждого приложения две системные очереди: очередь данных, поступающих к приложению из сети, и очередь данных, отправляемых этим приложением в сеть. Такие системные очереди называются **портами**[1], причем входная и выходная очереди одного приложения рассматриваются как один порт. Для идентификации портов им присваивают номера.

Если процессы представляют собой популярные системные службы, такие как FTP, telnet, HTTP, TFTP, DNS и т. п., то за ними закрепляются **стандартные назначенные номера**, называемые также **хорошо известными** (well-known) номерами портов. Эти номера закрепляются и публикуются в стандартах интернета (RFC 1700, RFC 3232). Так, номер 21 закреплен за серверной частью службы удаленного доступа к файлам FTP, а номер 23 — за серверной частью службы удаленного управления telnet. Назначенные номера из диапазона от 0 до 1023 являются *уникальными в пределах интернета* и закрепляются за приложениями *централизованно*.

Для тех приложений, которые еще не стали столь распространенными, номера портов назначаются *локально* разработчиками этих приложений или ОС в ответ на поступление запроса от приложения. На каждом компьютере ОС ведет список занятых и свободных номеров портов. При поступлении запроса от приложения, выполняемого на данном компьютере, ОС выделяет ему первый свободный номер. Такие номера называют **динамическими**. В дальнейшем все сетевые приложения должны адресоваться к данному приложению с указанием назначенного ему динамического номера порта. После того как приложение завершит работу, номер его порта возвращается в список свободных и может быть назначен другому приложению. Динамические номера являются *уникальными в пределах каждого компьютера*, при этом обычной является ситуация совпадения номеров портов приложений, выполняемых на разных компьютерах. Как правило, клиентские части известных приложений (DNS, WWW, FTP, telnet и др.) получают динамические номера портов от ОС.

Все, что сказано о портах, в равной степени относится и к обоим протоколам транспортного уровня (TCP и UDP). В принципе, нет никакой зависимости между назначением номеров портов для приложений, использующих протокол TCP, и приложений, работающих с протоколом UDP. Приложения, передающие данные на уровень IP по протоколу UDP, получают номера, называемые **UDP-портами**. Аналогично приложениям, обращающимся к протоколу TCP, выделяются **TCP-порты**. В том и другом случаях это могут быть как назначенные, так и динамические номера. Диапазоны чисел, из которых выделяются номера TCP- и UDP-портов, совпадают: от 0 до 1023 для назначенных и от 1024 до 65 535 для динамических. Однако никакой связи между назначенными номерами TCP- и UDP-портов нет. Даже если номера TCP- и UDP-портов совпадают, они идентифицируют разные приложения. Например, одному приложению может быть назначен TCP-порт 1750, а другому — UDP-порт 1750. В некоторых случаях, когда приложение может обращаться по выбору к протоколу TCP или UDP (например, таким приложением является DNS), ему, исходя из удобства запоминания, назначаются совпадающие номера TCP- и UDP-портов (в данном примере это *хорошо известный* номер 53).

---

[1] Порты приложений не надо путать с портами (сетевыми интерфейсами) оборудования.

Сокеты

Стандартные назначенные номера портов уникально идентифицируют тип приложения (FTP, или HTTP, или DNS и т. д.), но не могут использоваться для однозначной идентификации прикладных процессов, связанных с каждым из этих типов приложений. Пусть, например, на одном хосте запущены две *копии* DNS-сервера — DNS-сервер 1, DNS-сервер 2 (рис. 15.2). Каждый из этих DNS-серверов имеет хорошо известный UDP-порт 53. Какому из этих серверов нужно направить запрос клиента, если в DNS-запросе в качестве идентификатора сервера был указан только номер порта?

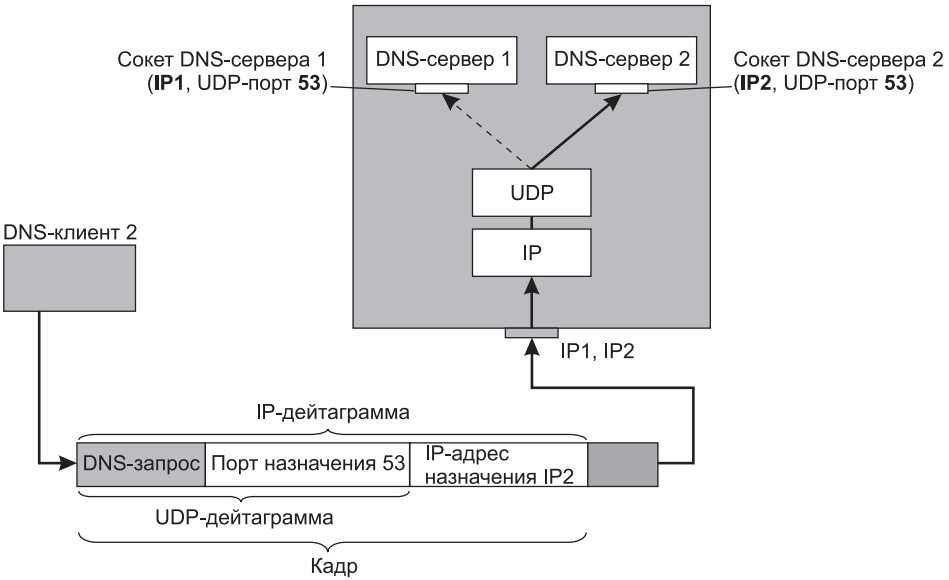


Рис. 15.2. Демультиплексирование протокола UDP на основе сокетов

Чтобы снять неоднозначность в идентификации приложений, разные копии связываются с разными IP-адресами. Для этого сетевой интерфейс компьютера, на котором выполняется несколько копий приложения, должен иметь соответствующее число IP-адресов — на рисунке это IP1 и IP2. Во всех IP-пакетах, направляемых DNS-серверу 1, в качестве IP-адреса указывается IP1, а DNS-серверу 2 — адрес IP2. Поэтому показанный на рисунке пакет, в поле данных которого содержится UDP-дейтаграмма с указанным номером порта 53, а в поле заголовка задан адрес IP2, однозначно будет направлен заданному адресату — DNS-серверу 2.

Прикладной процесс однозначно определяется в пределах сети и в пределах отдельного компьютера парой (IP-адрес, номер порта), называемой **сокетом** (socket).

Сокет, определенный IP-адресом и номером UDP-порта, называется **UDP-сокетом**, а IP-адресом и номером TCP-порта — **TCP-сокетом**.

Здесь следует уточнить описанную в предыдущих главах упрощенную картину прохождения пакета вверх по стеку. Действительно, после получения IP-пакета от протокола канального уровня протокол IP анализирует содержимое заголовка этого пакета, после чего заголовок отбрасывается, а «наверх» передается содержимое поля данных IP-пакета, например UDP-дейтаграмма. Упрощение состоит в том, что вместе с содержимым поля данных на транспортный уровень передается извлеченный из заголовка IP-адрес назначения, который и используется для однозначной идентификации приложения.

Протокол UDP и UDP-дейтаграммы

Протокол UDP, подобно IP, является дейтаграммным протоколом, реализующим так называемый ненадежный сервис *по возможности*, который не гарантирует доставку сообщений адресату. При работе на хосте-отправителе данные от приложений поступают протоколу UDP через порт в виде сообщений (рис. 15.3). Протокол UDP добавляет к каждому отдельному сообщению свой 8-байтный заголовок, формируя из этих сообщений собственные протокольные единицы, называемые **UDP-дейтаграммами**, и передает их нижележащему протоколу IP. В этом и заключаются его функции по *мультиплексированию* данных.

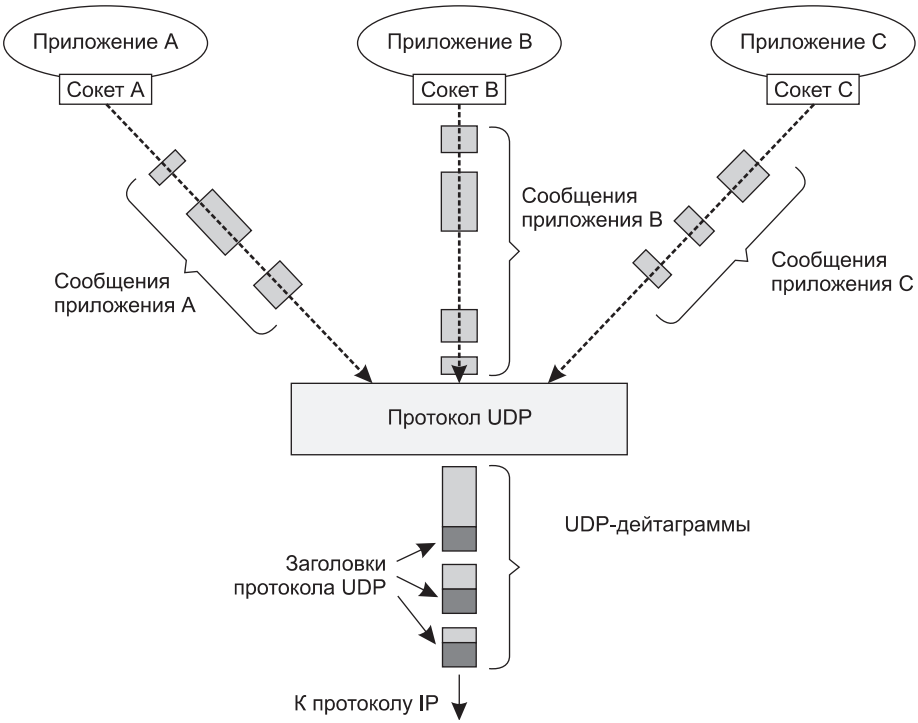


Рис. 15.3. Работа протокола UDP на хосте-отправителе

Каждая дейтаграмма переносит *отдельное пользовательское сообщение*. Сообщения могут иметь разную длину, не превышающую, однако, длину поля данных протокола IP, которое,

в свою очередь, ограничено размером кадра технологии нижнего уровня. Поэтому если буфер UDP переполняется, то сообщение приложения отбрасывается.

Заголовок UDP состоит из четырех двухбайтных полей:

- номер UDP-порта отправителя;
- номер UDP-порта получателя;
- контрольная сумма;
- длина дейтаграммы.

Приведем пример заголовка UDP с заполненными полями:

```
Source Port = 0x0035
Destination Port = 0x0411
Total length = 132 (0x84) bytes
Checksum = 0x5333
```

В этой UDP-дейтаграмме в поле данных, длина которого, как следует из заголовка, равна (132 – 8) байт, помещено сообщение DNS-сервера, что можно видеть по номеру порта источника (`Source Port = 0x0035`). В шестнадцатеричном формате это значение равно стандартному номеру порта DNS-сервера — 53.

Судя по простоте заголовка, протокол UDP несложен. Действительно, его функции сводятся к простой передаче данных между прикладным и сетевым уровнями, а также примитивному контролю искажений в передаваемых данных. При контроле искажений протокол UDP только *диагностирует, но не исправляет ошибку*. Если контрольная сумма показывает, что в поле данных UDP-дейтаграммы произошла ошибка, то протокол UDP просто отбрасывает поврежденную дейтаграмму.

Работая на хосте-получателе, протокол UDP принимает от протокола IP извлеченные из пакетов UDP-дейтаграммы. Полученные из IP-заголовка IP-адрес назначения и из UDP-заголовка номер порта используются для формирования UDP-сокета, однозначно идентифицирующего приложение, которому направлены данные. Протокол UDP освобождает дейтаграмму от UDP-заголовка. Полученное в результате сообщение он передает приложению на соответствующий UDP-сокет. Таким образом, протокол UDP выполняет *демультиплексирование* на основе сокетов.

Протокол TCP и TCP-сегменты

Протокол TCP предназначен для передачи данных между приложениями. Этот протокол основан на *логическом соединении*, что позволяет ему обеспечивать гарантированную доставку данных, используя в качестве инструмента ненадежный дейтаграммный сервис протокола IP.

При работе на хосте-отправителе протокол TCP рассматривает информацию, поступающую к нему от прикладных процессов, как *неструктурированный поток байтов* (рис. 15.4). Поступающие данные буферизуются средствами TCP. Для передачи на сетевой уровень из буфера «вырезается» некоторая непрерывная часть данных, которая называется **сегментом**[1] и снабжается заголовком.

[1] Заметим, что сегментом называют как единицу передаваемых данных в целом (поле данных и заголовок протокола TCP), так и отдельно поле данных.

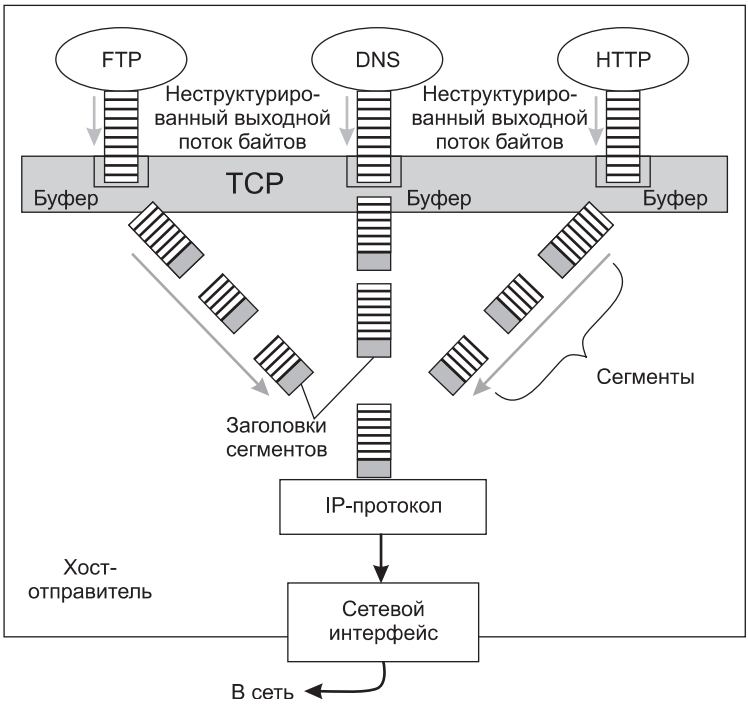


Рис. 15.4. Формирование TCP-сегментов из потока байтов

ПРИМЕЧАНИЕ

В отличие от протокола UDP, создающего дейтаграммы на основе логически обособленных единиц данных (сообщений, генерируемых приложениями), протокол TCP делит поток данных на сегменты без учета их смысла или внутренней структуры.

Заголовок TCP-сегмента содержит значительно больше полей, чем заголовок UDP, что отражает более развитые возможности протокола TCP (рис. 15.5). Краткие описания большинства полей помещены на рисунке (более подробно они рассматриваются в ходе изучения функций протокола TCP).

Коротко поясним значение однобитных полей, называемых **флагами** или **кодовыми битами** (code bits). Они расположены сразу за резервным полем и содержат служебную информацию о типе данного сегмента. Положительное значение сигнализируется установкой этих битов в единицу:

- **URG** — срочное сообщение;
- **ACK** — квитанция на принятый сегмент;
- **PSH** — запрос на отправку сообщения без ожидания заполнения буфера;
- **RST** — запрос на сброс соединения;
- **SYN** — сообщение, используемое для синхронизации счетчиков переданных данных при установлении соединения;

- **FIN** — признак достижения передающей стороной последнего байта в потоке передаваемых данных.



Рис. 15.5. Формат заголовка TCP-сегмента

Логические соединения — основа надежности TCP

Протокол транспортного уровня TCP предназначен для обеспечения надежной передачи данных в условиях, когда базовый сетевой уровень ненадежен. Для этого протокол TCP использует метод продвижения данных с установлением *логического соединения*. Как отмечалось, логическое соединение дает возможность участникам обмена следить за тем, чтобы данные не были потеряны, искажены или продублированы, а также чтобы они пришли к получателю в том порядке, в котором были отправлены. Эти задачи решаются путем комбинации разных подходов: нумерации сообщений, отслеживания квитанций, установки таймеров и повторных передач.

На рис. 15.6 показаны сети, соединенные маршрутизаторами, на которых установлен протокол IP. Протокол TCP устанавливает логические соединения между *прикладными процессами*, причем в каждом соединении участвуют только *два* процесса. TCP-соединение

является *дуплексным*, то есть каждый из участников этого соединения может одновременно получать и отправлять данные.

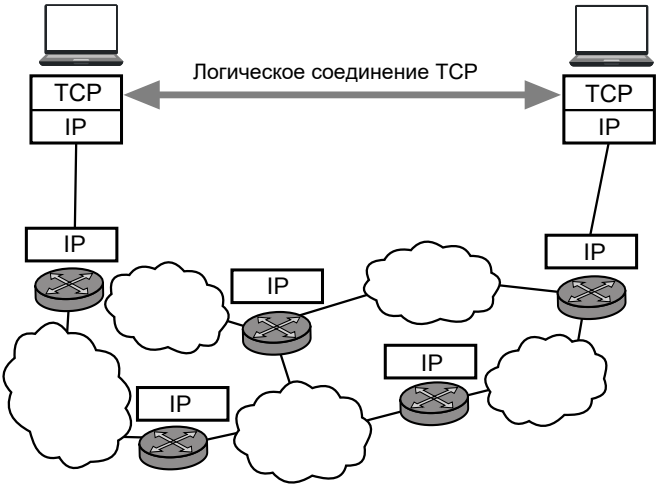


Рис. 15.6. TCP-соединение создает надежный логический канал между конечными узлами

При установлении логического соединения модули TCP договариваются между собой о параметрах процедуры обмена данными. В протоколе TCP каждая сторона соединения посылает противоположной стороне следующие параметры:

- *максимальный размер сегмента*, который она готова принимать;
- *максимальный объем данных* (возможно, несколько сегментов), которые она разрешает другой стороне передавать в свою сторону, даже если та еще не получила квитанцию на предыдущую порцию данных (размер **окна приема**);
- *начальный порядковый номер байта*, с которого она начинает отсчет потока данных в рамках данного соединения.

В результате переговорного процесса модулей TCP с двух сторон соединения определяются параметры соединения. Одни из них остаются постоянными в течение всего сеанса связи, другие — адаптивно изменяются.

Соединение устанавливается по инициативе клиентской части приложения. При необходимости выполнить обмен данными с серверной частью приложение-клиент обращается к нижележащему протоколу TCP, который в ответ на это обращение посылает сегмент-запрос на установление соединения протоколу TCP, работающему на стороне сервера (рис. 15.7, *а*). В числе прочего в запросе содержится флаг SYN, установленный в 1.

Получив запрос, модуль TCP на стороне сервера пытается создать «инфраструктуру» для обслуживания нового клиента. Он обращается к ОС с просьбой о выделении определенных системных ресурсов для организации буферов, таймеров, счетчиков. Эти ресурсы закрепляются за соединением с момента создания и до момента разрыва. Если на стороне сервера все необходимые ресурсы были получены и все необходимые действия выполнены, то модуль TCP посылает клиенту сегмент с флагами ACK и SYN. В ответ клиент посылает сегмент с флагом ACK и переходит в состояние установленного логического соединения

(состояние ESTABLISHED). Получив флаг ACK, сервер также переходит в состояние ESTABLISHED. На этом процедура установления соединения заканчивается, и стороны могут переходить к обмену данными. Соединение может быть разорвано в любой момент по инициативе любой стороны. Для этого клиент и сервер должны обменяться сегментами FIN и ACK в последовательности, показанной на рис. 15.7, *б* (здесь инициатором является клиент). Соединение считается закрытым по прошествии некоторого времени, в течение которого сторона-инициатор убеждается, что ее завершающий сигнал ACK дошел нормально и не вызвал никаких аварийных сообщений со стороны сервера.

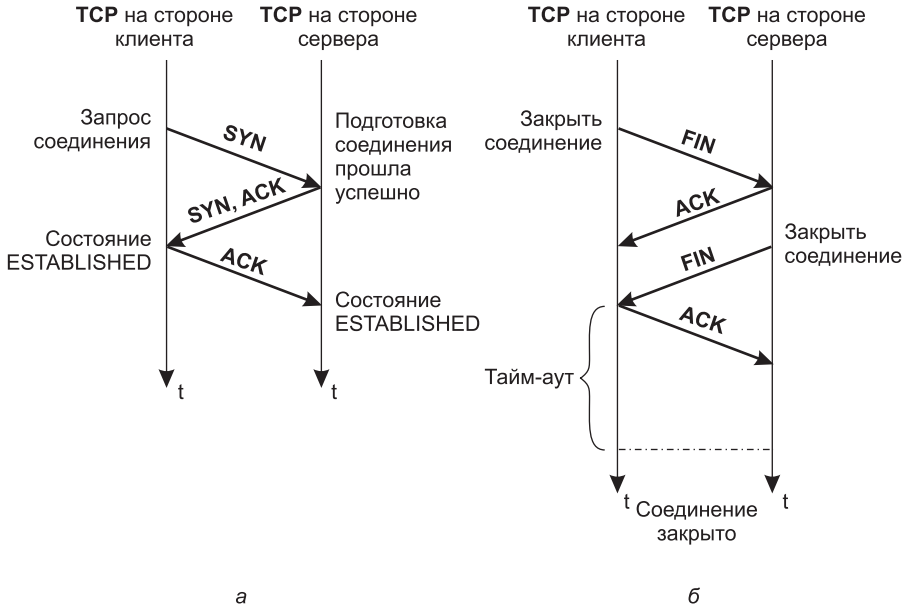


Рис. 15.7. Процедура установления и разрыва логического соединения при нормальном течении процесса

ПРИМЕЧАНИЕ

Мы описали процедуры установления и закрытия соединения очень схематично. Реальные протокольные модули работают в соответствии с более сложными алгоритмами, учитывающими всевозможные «нештатные» ситуации, такие, например, как задержки и потери сегментов, недостаточность ресурсов или неготовность сервера к установлению соединения. Кроме того, мы проигнорировали тот факт, что еще на этапе установления соединения стороны договариваются о некоторых параметрах своего взаимодействия, например, о начальных номерах посылаемых ими байтов. Вернемся к этим важным деталям работы протокола TCP чуть позже.

Логическое TCP-соединение однозначно идентифицируется парой сокетов, определенных для этого соединения двумя взаимодействующими процессами.

Сокет одновременно может участвовать в нескольких соединениях. Так, на рис. 15.8 показаны три компьютера (с адресами IP1, IP2, IP3), на каждом из них выполняется по одному приложению — APPL1, APPL2 и APPL3, сокеты которых соответственно (IP1, *n*1), (IP2, *n*2), (IP3, *n*3), а номера TCP-портов приложений — *n*1, *n*2, *n*3.

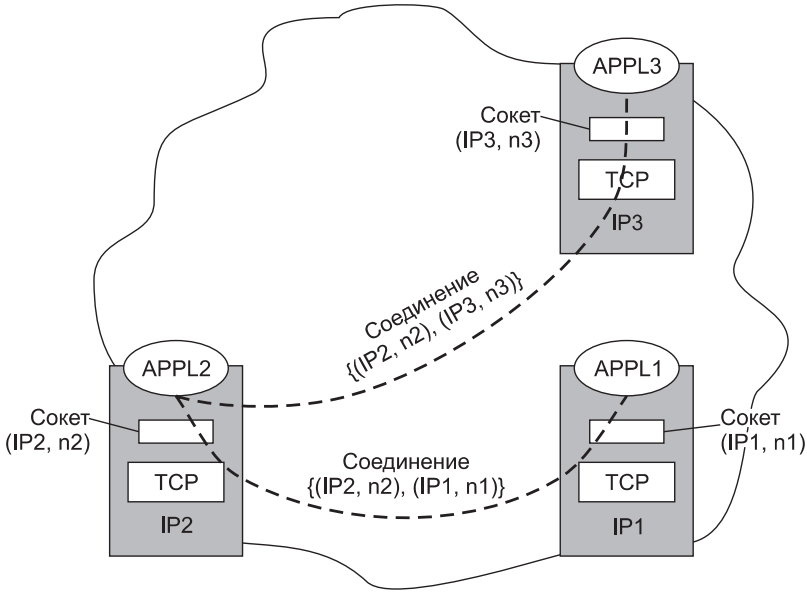


Рис. 15.8. Один сокет может участвовать в нескольких соединениях

На рисунке показаны два логических соединения, которые установило приложение 2 с приложением 1 и приложением 3. Логические соединения идентифицируются как {(IP2, *n*2), (IP1, *n*1)} и {(IP2, *n*2), (IP3, *n*3)} соответственно. Как видим, в обоих соединениях участвует один и тот же сокет — (IP2, *n*2). Теперь рассмотрим на примере, как протокол TCP выполняет демультиплексирование. Пусть некий поставщик услуг оказывает услугу по веб-хостингу, то есть на его компьютере клиенты могут разворачивать свои веб-серверы. Работа веб-сервера основана на протоколе прикладного уровня HTTP, который передает свои сообщения в TCP-сегментах. Модуль TCP ожидает запросы от веб-клиентов (браузеров), прослушивая хорошо известный порт 80.

На рис. 15.9 показан вариант хостинга с двумя веб-серверами — сервером www1.model.ru, имеющим IP-адрес IP1, и сервером www2.tour.ru с адресом IP2. К каждому из них может обращаться множество клиентов, причем клиенты могут одновременно работать как с сервером www1, так и с сервером www2. Для каждой пары «клиент — сервер» протоколом TCP создается *отдельное логическое соединение*.

На рисунке показаны два браузера, имеющие соответственно сокеты (IP_k, n_k) и (IP_m, n_m). Пользователь браузера k обращается одновременно к серверам WWW1 и WWW2. Наличие отдельных соединений для работы с каждым из этих серверов обеспечивает не только надежную доставку, но и разделение информационных потоков — у пользователя никогда не возникает вопроса, каким сервером ему была послана та или иная страница. Одновременно с пользователем браузера k с сервером WWW2 работает пользователь браузера m. И в этом случае отдельные логические соединения, в рамках которых идет работа обоих пользователей, позволяют изолировать их информационные потоки. На рисунке показаны буферы, количество которых определяется не числом веб-серверов и не числом клиентов, а числом логических соединений. Сообщения в эти буферы направляются в зависимости

от значений сокетов как отправителя, так и получателя. Отсюда — вывод: протокол TCP осуществляет демультиплексирование информации, поступающей на прикладной уровень, на основе *соединений* процессов или, что одно и то же, на основе идентифицирующих эти процессы *пар сокетов*.

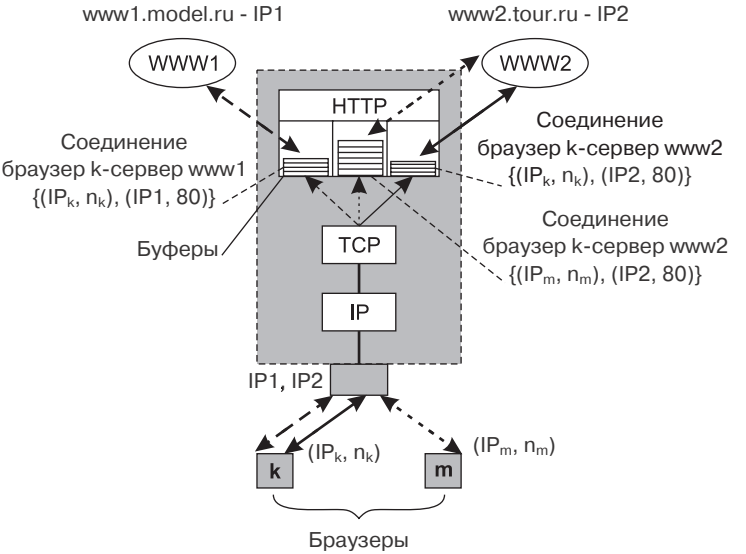


Рис. 15.9. Демультиплексирование данных протоколом TCP

Методы квитирования

Один из наиболее естественных приемов, используемых для организации надежного обмена данными, — передача с *квитированием*, суть которой состоит в следующем: отправитель отсылает данные, а получатель подтверждает их получение квитанциями в адрес отправителя. Если отправитель вовремя не получает квитанции на переданные данные, то он передает их *повторно*.

При всей простоте сформулированной схемы, за которой закрепилось особое название **«запрос повторной передачи»** (Automatic Repeat reQuest, **ARQ**), любая попытка ее реализации обрастает множеством деталей и вопросов. Например, должен ли отправитель ждать, пока не придет квитанция на отправленный пакет[1], прежде чем отсылать следующий? Должна ли принимающая сторона подтверждать приход каждого или сразу нескольких пакетов? Какое время ожидания квитанции источником является предельно допустимым? Что, если квитанция потеряется и отправитель еще раз пошлет тот же пакет? Каким образом приемник должен распознавать дубликаты пакетов, а источник — квитанций? Различные протоколы по-разному отвечают на эти вопросы, и в результате различаются производительностью, надежностью и объемом потребляемых ресурсов.

[1] В данном случае не имеет значения, какое название используется для единицы передаваемых данных — кадр, пакет или сообщение.

Все многообразие решений можно разделить на два класса:

- методы **простоя источника** (Stop-and-Wait);
- методы **скользящего окна**, которые свою очередь тоже подразделяются на два класса:
  - методы, использующие *окно передачи*, — к ним, в частности, относится метод **передачи с возвращением на N пакетов** (Go-Back-N);
  - методы, использующие *окно передачи и окно приема*, — примером является метод **передачи с выборочным повторением** (Selective Repeat).

Предполагается, что все эти методы работают в следующих условиях:

- Отправитель и получатель, в общем случае работающие *асинхронно*, осуществляют передачу пакетов по *ненадежной* линии связи, в которой возможны искажения, большие задержки и потери пакетов.
- Отправитель принимает данные от *протокола верхнего уровня* (приложения), получатель считывает их из приемного буфера и передает полученные данные на верхний уровень (приложению).
- Получатель располагает механизмом определения искаженных пакетов, например, по контрольной сумме.
- После успешного получения пакета получатель посылает отправителю квитанции (acknowledgment, **ACK**).
- Для отслеживания задержек пакетов используется **таймер**, **интервал тайм-аута** которого устанавливается равным предельному времени ожидания квитанции.

Метод простоя источника

В методе простоя источника отправитель передает последовательность ненумерованных пакетов. Метод требует, чтобы отправитель дожидался от получателя квитанции и только после этого посылал следующий пакет.

С переданным пакетом отправитель связывает таймер. Если установленный на таймере интервал истек, а квитанция не пришла, то фиксируется событие — **тайм-аут**: пакет (или квитанция на него) считается утерянным или искаженным, и передача пакета повторяется. На рис. 15.10 показано, что второй пакет (здесь пакеты нумеруются только для нашего удобства, отправитель и получатель не имеют дела с номерами) отсылается только после того, как пришла квитанция, подтверждающая доставку первого пакета. Однако затем произошла длительная пауза в отправке очередного, третьего пакета. В течение этой паузы источник, не дождавшись квитанции, которая была *утеряна*, послал повторно пакет 2, в результате получатель теперь имеет и оригинальный пакет 2, и его дубликат. Понятно, что при таком алгоритме работы принимающая сторона должна уметь распознавать дублированные пакеты и отбрасывать их.

Если первая квитанция не была утеряна, а просто шла к отправителю слишком долго, то возможна коллизия с приходом в источник двух квитанций на пакет-оригинал и пакет-дубль. Вторую квитанцию отправитель может интерпретировать как квитанцию на получение следующего по порядку пакета. Таким образом, отправитель также должен иметь возможность распознавать дубликаты квитанций.

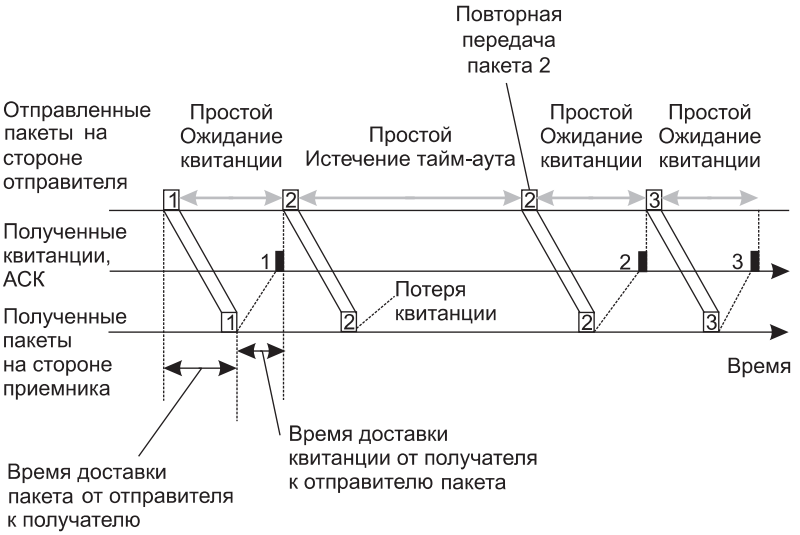


Рис. 15.10. Метод простоя источника

Частичное решение задачи распознавания дубликатов может быть достигнуто за счет включения в заголовок пакета *специального бита*. Этот бит устанавливается отправителем так, что ноль и единица в качестве его значения чередуются в пределах всей последовательности отправляемых пакетов. Приемник проверяет значение бита: если у двух последовательно пришедших пакетов значения данного бита равны двум единицам или двум нулям, то эти пакеты считаются дубликатами. Аналогично поступает отправитель с квитанциями. Такой способ распознавания дубликатов не является абсолютно надежным, поскольку основывается на предположении о том, что вероятность прихода в приемник дубликатов друг за другом достаточно высока. Очевидно и то, что данный метод ведет к очень низкому коэффициенту использования линии связи — ведь основную часть времени передатчик простаивает в ожидании прихода квитанции.

Концепция скользящего окна

Концепция **скользящего окна** (sliding window) заключается в том, что для повышения скорости передачи данных отправителю разрешается передать некоторое количество пакетов, *не дожидаясь прихода на эти пакеты квитанций*.

Так как в этом методе одновременно могут существовать несколько неподтвержденных пакетов, то их необходимо каким-то образом различать, чтобы отправитель мог понять, получение какого пакета подтверждает квитанция. Для идентификации пакетам присваиваются *уникальные последовательные номера*, размещаемые в заголовках пакетов. Разрядность поля «Номер пакета» определяет диапазон возможных номеров. Когда этот диапазон исчерпывается, нумерация пакетов снова начинается с нуля. Таким образом, нельзя абсолютно исключить ситуацию, когда в сети существуют пакеты с одинаковыми

номерами (позже мы еще вернемся к этому вопросу). Окно определяется на последовательности пронумерованных пакетов (рис. 15.11). Оно всегда имеет нижнюю границу, называемую также **базой окна** (здесь это пакет с номером 9), и верхнюю границу (14). Количество номеров, попадающих в пределы окна, называют **размером окна** (6). Очевидно, что окно всегда перемещается в сторону бóльших номеров.

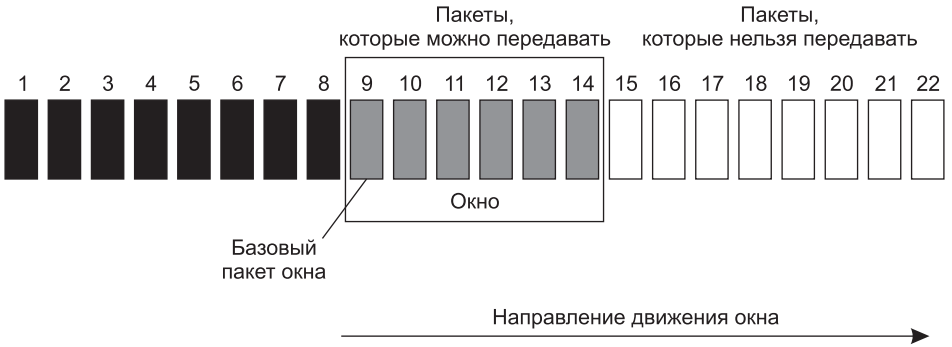


Рис. 15.11. Окно передачи

Окно, показанное на рисунке, регулирует процесс передачи — оно ограничивает количество пакетов, которые отправитель может передать до получения квитанции. Окно может быть определено не только для передачи, но и для приема пакетов (в таком случае ограничивается количество пакетов, которые получатель может принять, не передавая их на верхний уровень).

ПРИМЕЧАНИЕ

Может возникнуть вопрос, зачем вообще нужно это ограничение и почему нельзя разрешить отправителю передавать произвольное количество пакетов? Дело в том, что сняв ограничение и позволив тем самым существовать в сети большому количеству искаженных, потерявшихся и не получивших подтверждения пакетов, мы сделаем процесс передачи неконтролируемым, в результате, например, для хранения таких пакетов потребуется буфер неограниченных размеров.

На рис. 15.12 показана идеализированная схема «скольжения» окна передачи. В этом примере предполагается, что пакеты и квитанции не теряются и приходят в том же порядке, в котором были отправлены. При этом интервалы между пакетами и квитанциями являются неравномерными. Движение окна определяется, во-первых, поступлением квитанций — подтверждение успешного приема очередного пакета позволяет переместить окно вперед, во-вторых, исчерпанием окна — окно приостанавливается, когда отправитель передал все пакеты из окна, но не получил ни на один из них квитанции.

Здесь последовательность номеров пакетов отображается на оси ординат, а значит, окно скользит вдоль вертикали. Окно имеет размер 5. В начальный момент, когда еще не послано ни одного пакета, окно определяет диапазон номеров пакетов от 1 до 5 включительно. Источник начинает передавать пакеты и через какое-то время получает в ответ квитанции. Для простоты предположим, что квитанции поступают в той же последовательности (но не обязательно в том же темпе), что и пакеты, которым они соответствуют. В момент получения отправителем квитанции 1 окно сдвигается на одну позицию вверх, определяя новый диапазон разрешенных к отправке пакетов (от 2 до 6).

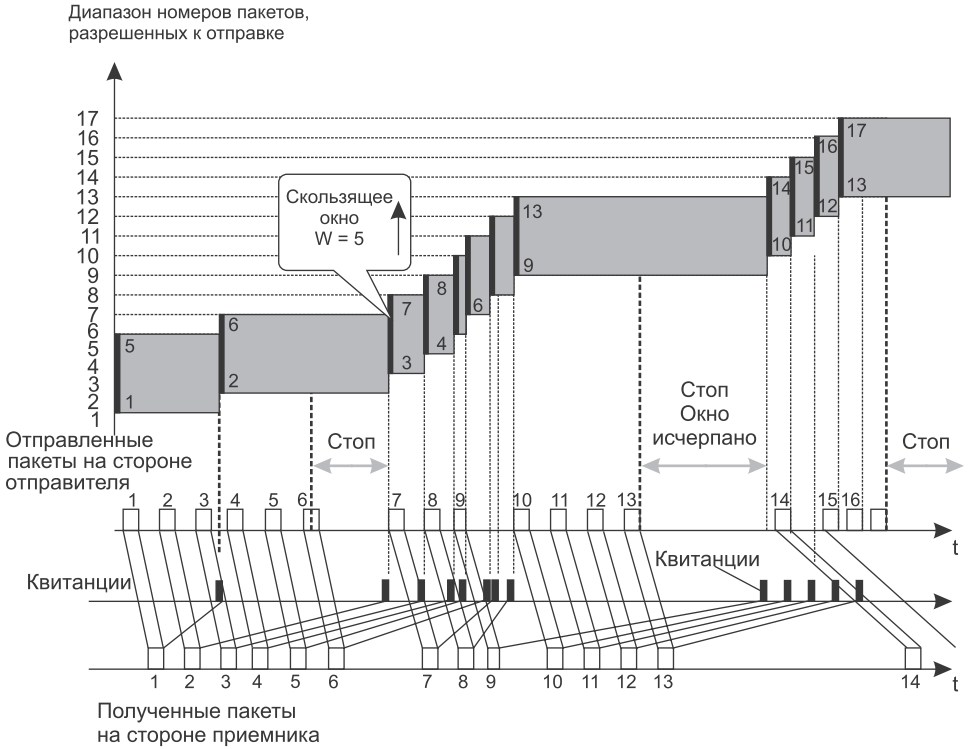


Рис. 15.12. Метод скользящего окна

Процессы отправки пакетов и получения квитанций идут независимо друг от друга. В нашем примере отправитель продолжает передавать пакеты, но некоторое время не получает на них квитанции. После передачи пакета 6 окно исчерпывается, и источник приостанавливает передачу. После получения квитанции 2 (на пакет 2) окно сдвигается вверх на единицу, определяя диапазон разрешенных к передаче пакетов от 3 до 7. Аналогичное «скольжение» окна вверх происходит после получения каждой квитанции: окно сдвигается вверх на 1, но его размер при этом не меняется. После прихода квитанции 8 окно оказывается в диапазоне от 9 до 13 и остается таковым достаточно долго, так как по каким-то причинам источник перестает получать подтверждения о доставке пакетов. Отправив последний разрешенный пакет 13, передатчик снова прекращает передачу, с тем чтобы возобновить ее после прихода квитанции 9.

Теперь, когда мы познакомились с концепцией скользящего окна, рассмотрим методы передачи, построенные на ее основе.

Передача с возвращением на N пакетов

В данном методе ставится задача обеспечить более эффективное использование линии связи, чем в методе с простоем источника. Для этого отправителю разрешается отправлять следующие пакеты, не дожидаясь подтверждения получателем их успеш-

ного приема. Количество переданных таким образом пакетов ограничивается окном. Поскольку отправитель должен иметь возможность при необходимости повторить передачу любого из переданных пакетов, их необходимо буферизовать. Кроме того, отправитель должен отслеживать статус пакетов (отправлен/не отправлен, подтвержден/не подтвержден). На рис. 15.13 показано положение окна передачи в некоторый момент времени. Базой окна в данном случае является номер 7, а верхней границей — номер 17. Все пакеты с номерами в этом диапазоне отправитель вправе передавать независимо от поступления квитанций. С левой стороны окна находятся более старые пакеты (1–6), которые были переданы и подтверждены. Пакеты (18–22) справа от окна в данный момент передавать запрещено.



Рис. 15.13. Окно передачи в методе с возвратом на N пакетов

В пределах окна имеются как уже отправленные, но не подтвержденные пакеты (7–12), так и не отправленные пакеты, которые разрешено отправлять (13–17). Как только самый старый пакет в окне (7) получит подтверждение успешного приема, окно сдвинется направо на единицу.

Рассмотрим *алгоритм работы получателя*. При поступлении нового пакета получатель всегда выполняет одни и те же действия, проверяя:

- является ли пакет неискаженным;
- является ли он следующим по порядку в последовательности уже полученных пакетов.

Если оба условия выполнены, то пакет принимается и передается на более высокий уровень (размер окна приема в этом методе равен 1), а отправителю посылается квитанция, в которой указывается номер успешно принятого пакета. Если же хотя бы одно из этих условий не выполнено, то пакет *отбрасывается*, а отправителю снова посылается квитанция с номером последнего по времени успешно принятого пакета.

Отметим очень важный момент: в том случае, когда получатель принимает и подтверждает прием пакетов в строгом соответствии с их порядковыми номерами (а именно с такой ситуацией мы и имеем дело), квитанция о любом принятом пакете говорит и о том, что *все* предыдущие пакеты также были приняты успешно. Такого рода квитанции называют *кумулятивными* или *накопительными*. Другими словами, нет необходимости дублировать потерянную кумулятивную квитанцию, потому что она компенсируется приходом следующей квитанции, также являющейся кумулятивной.

Теперь посмотрим на *алгоритм работы отправителя*, использующего *один* таймер, интервал тайм-аута которого устанавливается равным предельному времени ожидания квитанции, отправленной получателем для подтверждения успешности доставки и приема *базового пакета* окна. На процесс передачи пакетов влияют следующие события:

- *Исчерпание окна* — ситуация, когда все пакеты из окна отправлены, но не подтверждены. В этом случае *передача останавливается*.
- *Тайм-аут* — это событие интерпретируется отправителем как потеря пакета или квитанции, а значит, *выполняется повторная передача* базового пакета и для него *заново устанавливается таймер*. При этом, подчеркнем, повторно передается не только этот пакет, но и *все* переданные, но неподтвержденные пакеты. Именно такая реакция на недоставленный пакет и дала методу название «возвращение на N пакетов». Повторная передача всех неподтвержденных пакетов нужна, так как даже если приемник и получил их, он их отбросил, поскольку они не образовывали непрерывную последовательность пакетов.
- *Поступление квитанции*. Соответствующий пакет и все пакеты в пределах окна с меньшими номерами считаются успешно принятыми (учитывается кумулятивность квитанции). *Окно сдвигается*, фиксируется новый базовый пакет, переустанавливается таймер. Если до этого передача была остановлена из-за исчерпания окна, то *передача возобновляется*.
- Если квитанция на базовый пакет пришла после истечения его таймера, а значит, и после его повторной передачи, то эта квитанция «засчитывается» и выполняются все действия, определенные для этого случая.

Зная алгоритм работы отправителя, можно легко предсказать, что произойдет с преждевременно пришедшим и поэтому отброшенным пакетом: либо истечет таймер для одного из предшествующих пакетов и он вместе со всеми остальными будет передан повторно, либо сам этот пакет станет базовым и после непременного истечения его таймера (он ведь был отброшен) произойдет его повторная передача.

Подводя итог, отметим, что алгоритм работы получателя в этом методе существенно проще, чем отправителя. Получатель не использует окно, а следовательно, не нуждается в буферизации пакетов и отслеживании их статуса. От получателя требуется только распознавать ошибочные пакеты и отслеживать последовательность их номеров. Эффективность данного метода выше по сравнению с методом простоя источника за счет передачи в линию связи сразу нескольких пакетов. Но для него характерна *избыточность*: во-первых, получатель отбрасывает не только искаженный, но и корректно принятый пакет, если его номер выбивается из последовательности, а во-вторых, отправитель повторно передает не только потерянный или искаженный пакет, но и пакеты, отправленные после него.

Передача с выборочным повторением

В данном методе, как это следует из его названия, получатель может *выборочно* запросить повторную передачу отдельного пакета, а не всей последовательности переданных пакетов, как это происходит при передаче с возвращением на N пакетов.

Во избежание избыточных повторных передач принимающей стороне запрещено отбрасывать правильно принятый пакет лишь потому, что его номер выбивается из последо-

вательности. А это значит, что данный пакет, как и другие нарушающие последовательность пакеты, получатель должен где-то временно сохранять. Для этого организуется буфер, в котором хранятся и упорядочиваются прибывающие пакеты в соответствии с их номерами. Согласно поставленной задаче выборочного повторения ошибочных пакетов, выборочными (индивидуальными) должны быть и квитанции, позволяющие подтверждать успешность приема каждого отдельного пакета, и таймеры, замеряющие интервалы тайм-аутов для каждого отправленного пакета[1]. В данном методе концепция скользящего окна используется как для передачи, так и для приема пакетов. Оба окна определены на одной и той же последовательности номеров пакетов, а размеры окон *одинаковы*. Но их «скольжение» не является синхронным. В зависимости от возникающих ошибок при передаче пакетов и квитанций окно приема может опережать окно передачи. Отправитель и получатель работают только с пакетами, находящимися в пределах окна передачи и окна приема соответственно. На рис. 15.14 показано положение окон передачи и приема в некоторый момент времени.

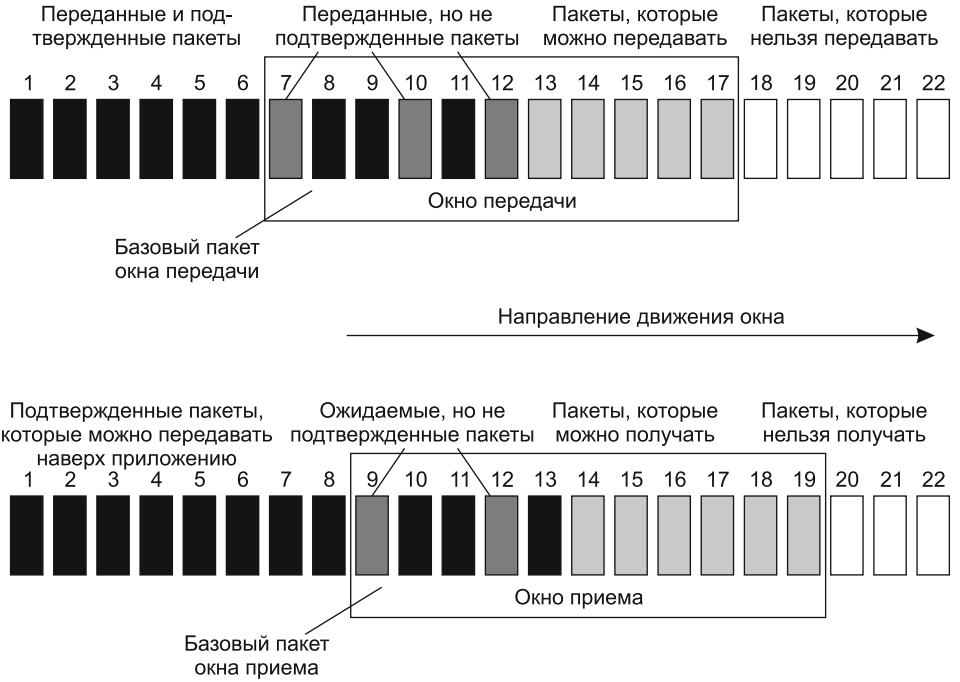


Рис. 15.14. Окна передачи и приема при выборочном повторении

Отправитель уже передал принятые от верхнего уровня пакеты 1–6, получил на них квитанции и ждет квитанцию на переданный пакет с номером 7. А получатель собрал и передал своему верхнему уровню непрерывную последовательность пакетов 1–8, но все еще не получил пакет 9. Окно передачи в данный момент позиционируется между базовым

[1] Обычно в протоколах используется множество программных таймеров, построенных на основе одного аппаратного.

пакетом 7 и граничным пакетом 17, а окно приема — в границах 9–19. Все пакеты с номерами, попадающими в окна, отправитель имеет право передавать, а получатель — принимать независимо от поступления квитанций.

Поскольку в этом методе квитанция является индивидуальной, а не накопительной, то в окне передачи среди переданных пакетов могут быть как подтвержденные (8, 9, 11), так и неподтвержденные (7, 10, 12). Номера из окна приема в общем случае могут соответствовать:

- принятым и подтвержденным пакетам (10, 11, 13), которые не могут быть переданы из буфера верхнему уровню, так как в их последовательность вклинились номера некоторых отсутствующих пакетов (9, 12);
- ожидаемым, еще не полученным пакетам (9, 12);
- пакетам, которые получателю разрешено принимать (14–19), так как их номера попадают в окно приема.

Получатель принимает, размещает в буфере и подтверждает квитанцией любой принятый пакет при условии, что он не искажен и его номер попадает в окно приема. Если принят базовый пакет, то левая граница окна приема сдвигается до первого ожидаемого, но еще не полученного пакета. На рисунке базовым является пакет 9. Когда он будет успешно принят, пробел с недостающим пакетом заполнится, окно сдвинется и новым базовым пакетом станет пакет 12. Выдвинувшаяся за границу окна приема непрерывная последовательность пакетов 9, 10, 11 тогда может быть передана верхнему уровню. На работе отправителя сказываются следующие события:

- *Исчерпание окна.* Отправитель последовательно посылает пакеты до тех пор, пока не исчерпается окно передачи.
- *Приход квитанции.* Отправитель, получив квитанцию, присваивает пакету статус успешно переданного. Если это был базовый пакет (например, на рисунке — пакет 7), то окно смещается вправо до первого по порядку принятого, но неподтвержденного пакета, который становится базой (на рисунке — пакет 10).
- *Тайм-аут.* Таймер устанавливается для каждого пакета отдельно, по истечении интервала тайм-аута соответствующий пакет повторяют. Таким образом, пакет повторяют, только если он был потерян или искажен. В некоторых протоколах после истечения таймера и повторной отправки пакета, следующий интервал тайм-аута выбирается в два раза большим, чем предыдущий, а затем, если таймер снова истечет, интервал снова удваивается и т. д. Вместо повторения передачи через равные промежутки времени, TCP каждый раз удлиняет в два раза время до следующей попытки и тем самым уменьшает количество повторно переданных дубликатов TCP.

Возможна и ситуация, в которой пакет благополучно был принят, но квитанция на него потерялась. Тогда к получателю придет дубликат пакета. Получатель не должен его игнорировать — ему следует подтвердить квитанцией прием дубликата, иначе отправитель «застрянет» на этом пакете, бесконечно повторяя его.

Подводя итог, можно отметить, что методы, использующие скользящее окно, сложнее в реализации, чем метод простоя источника, так как в первом случае требуется поддержание буфера (или буферов в случае выборочной передачи), отслеживание номеров и статуса пакетов, а также определение по меньшей мере двух важных параметров алгоритма: размер окна и величину интервала тайм-аута.

Метод скользящего окна в протоколе TCP

Сегменты и поток байтов

Алгоритм скользящего окна в протоколе TCP имеет некоторые существенные особенности. Так, в рассмотренном обобщенном алгоритме скользящего окна единицей передаваемых данных является пакет, и размер окна также определяется в пакетах, в то время как в протоколе TCP дело обстоит совсем по-другому. Хотя единицей передаваемых данных протокола TCP является сегмент (аналог пакета в данном контексте), окно определено на множестве *нумерованных* байтов *неструктурированного потока* данных, передаваемого приложением протоколу TCP.

В ходе переговорного процесса модули TCP обеих участвующих в обмене сторон договариваются между собой о параметрах процедуры обмена данными. Одни из них остаются постоянными в течение всего сеанса связи, другие — в зависимости, например, от интенсивности трафика и/или размеров буферов — адаптивно изменяются. Одним из таких параметров является *начальный номер байта*, с которого будет вестись отсчет в течение всего функционирования данного соединения. У каждой стороны — свой начальный номер. Нумерация байтов в пределах сегмента осуществляется, начиная от заголовка (рис. 15.15).

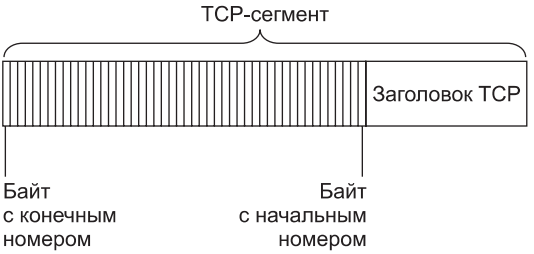


Рис. 15.15. Нумерация байтов в TCP-сегменте

Когда отправитель посылает TCP-сегмент, он помещает в поле *последовательного номера* номер первого байта данного сегмента, который служит *идентификатором* сегмента. На рис. 15.16 показано четыре сегмента размером 1460 байт и один — размером 870 байт. Идентификаторами этих сегментов являются номера 32600, 34060, 35520 и т. д. На основании этих номеров получатель TCP-сегмента не только отличает данный сегмент от других, но и позиционирует полученный фрагмент относительно общего потока байтов. Кроме того, он может сделать вывод, например, о том, что полученный сегмент является дубликатом или что между двумя полученными сегментами пропущены данные и т. д.

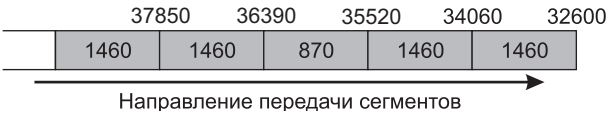


Рис. 15.16. Порядковый номер и номер квитанции

В качестве квитанции получатель сегмента отсылает ответное сообщение (сегмент), в поле *подтвержденного номера* которого он помещает число, на единицу превышающее макси-

мальный номер байта в полученном сегменте. Так, для первого отправленного сегмента, изображенного на рисунке, квитанцией о получении (подтвержденным номером) будет число 34060, для второго — 35520 и т. д. Подтвержденный номер часто интерпретируют не только как оповещение о благополучной доставке, но и как номер следующего ожидаемого байта данных. Квитанция в протоколе TCP посылается только в случае правильного приема данных (то есть нет возможности послать «отрицательную» квитанцию о неполучении сегмента). Таким образом, отсутствие квитанции означает либо потерю сегмента, либо потерю квитанции, либо прием искаженного сегмента. В соответствии с определенным форматом один и тот же TCP-сегмент может нести в себе как пользовательские данные (в поле данных), так и квитанцию (в заголовке), которой подтверждается получение данных от другой стороны.

Система буферов при дуплексной передаче

Поскольку протокол TCP является дуплексным, каждая сторона одновременно выступает и как отправитель, и как получатель. У каждой стороны есть пара буферов: один — для хранения принятых сегментов, другой — для сегментов, которые только еще предстоит отправить. Кроме того, имеется буфер для хранения копий сегментов, которые были отправлены, но квитанции о получении которых еще не поступили (рис. 15.17).

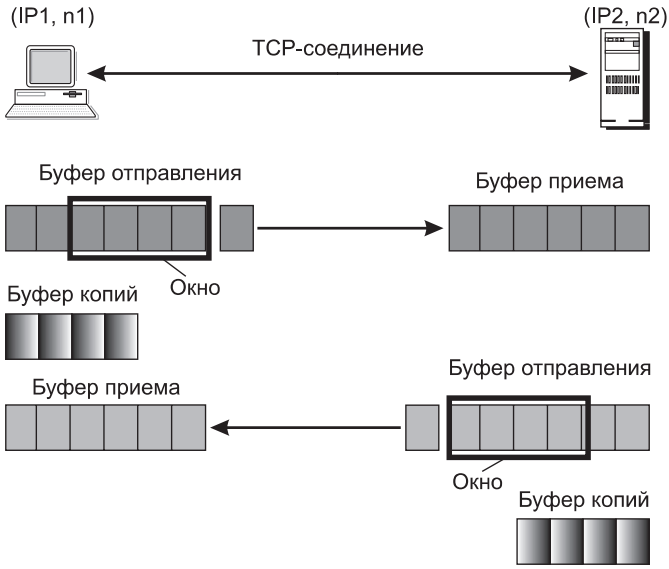


Рис. 15.17. Система буферов TCP-соединения

И при установлении соединения, и в ходе передачи обе стороны, выступая в роли получателя, посылают друг другу **окна приема** (receiver window). Каждая из сторон, получив окно приема, «узнает», сколько байтов ей разрешается отправить с момента получения последней квитанции. То есть, посылая окна приема, обе стороны пытаются регулировать поток байтов в свою сторону, сообщая своему «визави», какое количество байтов (начиная с номера байта, о котором уже была выслана квитанция) они готовы в настоящий момент

принять. На рис. 15.18 показан поток байтов, поступающий от приложения в выходной буфер модуля TCP.

Из потока байтов модуль TCP «нарезает» последовательность сегментов и поочередно отправляет их приложению-получателю. Для ясности на рисунке принято направление перемещения данных справа налево. В этом потоке можно указать несколько логических границ:

- Первая граница отделяет сегменты, которые уже были отправлены и на которые уже пришли квитанции. Последняя квитанция пришла на байт с номером *N*.
- По другую сторону этой границы располагается окно размером *W* байт. Часть байтов, входящих в окно, составляют сегменты, которые также уже отправлены, но квитанции на которые пока не получены.
- Оставшаяся часть окна — это сегменты, которые пока не отправлены, но могут быть отправлены, так как входят в пределы окна.
- И наконец, последняя граница указывает на начало последовательности сегментов, ни один из которых не может быть отправлен до тех пор, пока не придет очередная квитанция и окно не будет сдвинуто вправо.

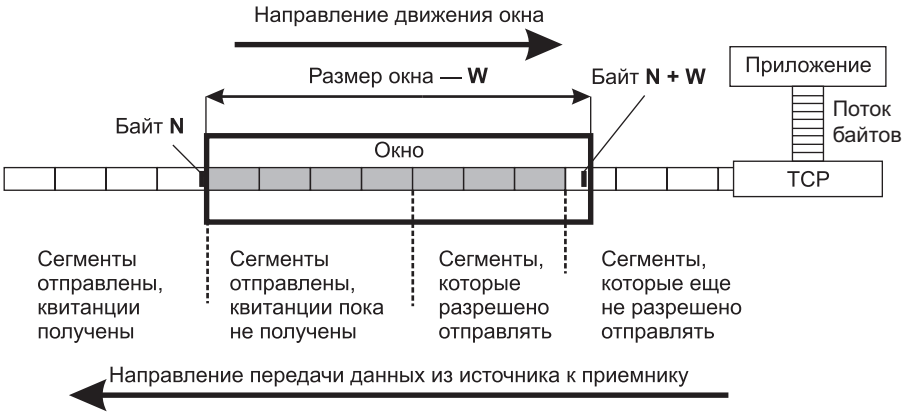


Рис. 15.18. Особенности реализации алгоритма скользящего окна в протоколе TCP

Накопительный принцип квитирования

Если размер окна равен *W*, а последняя по времени квитанция содержала значение *N*, то отправитель может посылать новые сегменты до тех пор, пока в очередной сегмент не попадет байт с номером *N* + *W*. Этот сегмент выходит за рамки окна, и передачу в таком случае необходимо приостановить до прихода следующей квитанции. Получатель может послать квитанцию, подтверждающую получение сразу нескольких сегментов, если они образуют непрерывный поток байтов. Например (рис. 15.19, *а*), если в буфер, плотно, без пропусков заполненный потоком байтов до 2354 включительно, поочередно поступили сегменты (2355–3816), (3817–5275) и (5276–8400), где цифры в скобках означают номера первых и последних байтов каждого сегмента, то получателю достаточно отправить только одну квитанцию на все три сегмента, указав в ней в качестве номера квитанции значение 8401. Таким образом, процесс квитирования в TCP является *накопительным*.

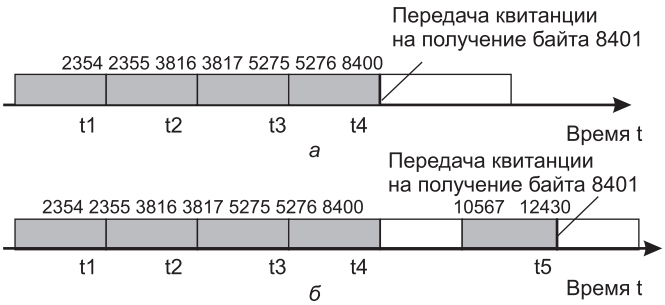


Рис. 15.19. Накопительный принцип квитирования: *а* — плотное заполнение буфера (в момент t4 передается квитанция на байт 8401), *б* — неплотное заполнение буфера (в момент t5 снова передается квитанция на байт 8401)

Вполне возможны ситуации, когда сегменты приходят к получателю не в том порядке, в котором были посланы, то есть в приемном буфере может образоваться «прогалина» (рис. 15.19, *б*). Пусть, к примеру, после указанных ранее трех сегментов вместо следующего по порядку сегмента (8401–10566) пришел сегмент (10567–12430). Очевидно, что послать в качестве номера квитанции значение 12431 нельзя, потому что это означало бы, что получены все байты вплоть до 12430. Поскольку в потоке байтов образовался разрыв, получатель может только еще раз повторить квитанцию 8401, говоря тем самым, что все еще ожидает поступления потока байтов, начиная с 8401, то есть подтверждает получение не отдельных блоков данных, а непрерывной последовательности байтов. Таким образом, повторная квитанция имеет как положительный смысл — «все байты до номера 8401 получены», так и отрицательный смысл — «некоторое количество байтов, начиная с номера 8401, не получено».

Когда протокол TCP передает в сеть сегмент, он «на всякий случай» помещает его копию в буфер, называемый также *очередью повторной передачи*, и запускает таймер. Когда приходит квитанция на этот сегмент, соответствующая копия удаляется из очереди. Если же квитанция не приходит до истечения срока, то сегмент, вернее, его копия, посылается повторно. Может случиться и так, что копия сегмента придет, когда исходный сегмент уже окажется на месте, — тогда дубликат попросту отбрасывается.

В некоторых версиях TCP [RFC 2018] от получателя не требуется следовать накопительному принципу подтверждения, в соответствии с которым квитанция посылается на последний правильно принятый сегмент, пришедший в правильном порядке. TCP-получатель *выборочно* подтверждает сегменты, пришедшие с нарушением порядка, а TCP-отправитель работает в режиме выборочной повторной передачи, пропуская повторную передачу сегментов, которые уже были выборочно подтверждены получателем.

Управление потоком и управление перегрузками

Параметры процесса управления потоком

Теперь, когда мы знакомы с алгоритмами квитирования, особый интерес приобретает способ выбора значений параметров этих алгоритмов, и главные из них — размер окна приема и значение интервала тайм-аута.

Размер окна приема связан с наличием в данный момент свободного места в буфере принимающей стороны. При его переполнении модуль TCP, отправляя квитанцию, помещает в нее новый уменьшенный размер окна, которым теперь должна руководствоваться отправляющая сторона. Если он совсем отказывается от приема, то в квитанции указывается *окно нулевого размера*. Однако даже после этого приложение может послать сообщение на отказавшийся от приема порт. Для этого сообщение должно сопровождаться *указателем срочности URG*. В такой ситуации порт обязан принять сегмент, даже если для этого придется вытеснить из буфера уже находящиеся там данные. После приема квитанции с нулевым значением окна протокол-отправитель время от времени делает контрольные попытки продолжить обмен данными. Если протокол-приемник уже готов принимать информацию, то в ответ на контрольный запрос он посылает квитанцию с указанием ненулевого размера окна.

В общем случае окна приема на разных концах соединения имеют разный размер. Например, можно ожидать, что сервер, вероятно, обладающий большим буфером, пошлет клиентской станции окно приема большее, чем клиент серверу. В зависимости от состояния сети то одна, то другая стороны могут объявлять новые значения окон приема, динамически уменьшая и увеличивая их. Чем больше окно, тем бóльшая порция неподтвержденных данных может быть послана в сеть. Но если пришло большее количество данных, чем может быть принято модулем TCP, то данные отбрасываются. Это ведет к излишним пересылкам информации и ненужному росту нагрузки на сеть и модуль TCP.

В то же время малый размер окна может ограничить передачу данных скоростью, которая определяется временем путешествия по сети отдельного сегмента. Чтобы избежать применения малых окон, в некоторых реализациях TCP предлагается получателю данных откладывать уменьшение размеров окна до тех пор, пока свободное место не составит 20–40 % от максимально возможного объема памяти для этого соединения. Но и отправителю не стоит спешить с посылкой данных, пока окно принимающей стороны не станет достаточно большим. Учитывая эти соображения, разработчики протокола TCP предложили схему, согласно которой при установлении соединения заявляется большое окно, но впоследствии его размер существенно уменьшается. Существуют и другие, прямо противоположные, алгоритмы настройки окна, когда вначале выбирается минимальное окно, а затем, если сеть справляется с предложенной нагрузкой, его размер резко увеличивается.

Другой важный вопрос: каким на каждой из сторон должно быть выбрано предельное время ожидания (интервал тайм-аута) очередной квитанции? От ответа на этот вопрос зависит производительность протокола TCP. При выборе интервала тайм-аута должны учитываться скорость и надежность линий связи, их протяженность и многие другие факторы. Интервал тайм-аута не должен быть слишком коротким, чтобы по возможности исключить избыточные повторные передачи, снижающие полезную пропускную способность системы. Но он не должен быть и слишком длинным, чтобы избежать длительных простоев, связанных с ожиданием несуществующей или «заблудившейся» квитанции (рис. 15.20).

Понятно, что интервал тайм-аута для некоторого сегмента должен превосходить время оборота RTT этого сегмента. Это время не может быть вычислено раз и навсегда и зависит от состояния сети в каждый момент времени, поэтому в качестве оценки используются измерения для некоторой выборки сегментов. В протоколе TCP [RFC 6298] время оборота определяется с помощью *адаптивного* алгоритма, идея которого состоит в следующем. При каждой передаче засекается время от момента отправки сегмента до прихода квитанции о его приеме. Получаемые значения времени оборота усредняются с весовыми коэффи-

циентами, возрастающими от предыдущего замера к последующему. Это делается с тем, чтобы усилить влияние последних замеров. В качестве RTT выбирается среднее время оборота, умноженное на некоторый коэффициент. Практика показывает, что значение этого коэффициента должно превышать 2. В сетях с большим разбросом времени оборота при вычислении оценки RTT учитывается также дисперсия этой величины.

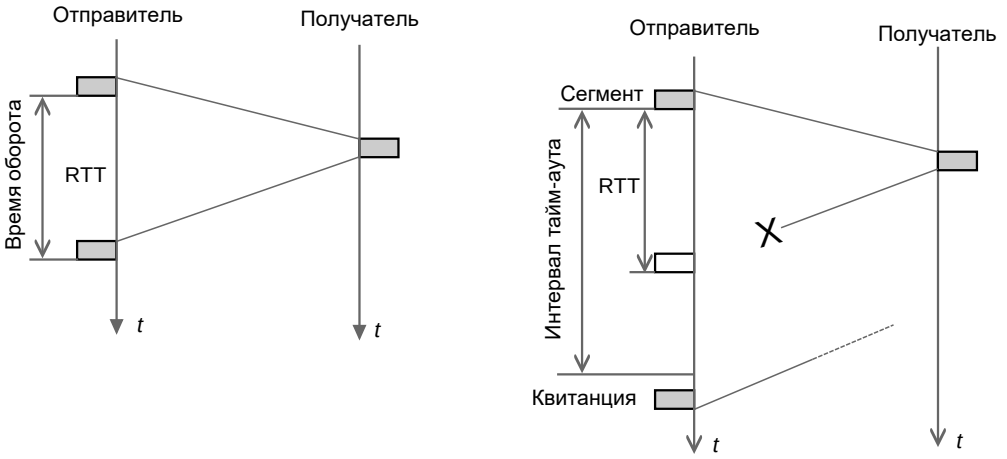


Рис. 15.20. Время оборота и интервал тайм-аута

Управление перегрузкой

Ранее, обсуждая протокол TCP, мы фокусировались на его возможностях по обеспечению надежной доставки данных между двумя прикладными процессами. Для этого TCP устанавливает логическое соединение и следит за тем, чтобы отправитель и получатель работали согласованно, чтобы буферы у них не переполнялись и как можно меньше пакетов отбрасывалось и повторно передавалось, чтобы не было необязательных простоев, — все эти действия по согласованию скорости отправки данных отправителем и скорости чтения их из буфера получателем относятся к **управлению потоком данных** (flow control) между приложениями.

Однако у протокола TCP есть и другая, отличная от управления потоком, задача — **управление перегрузками в сети** (congestion control)[1].

На работу TCP влияет степень загрузки сети: чем она выше, тем больше вероятность, что сегмент данных или квитанция потеряются. Но и работа TCP-соединения, хотя и косвенным образом, но влияет на загрузку сети: повторно посылаемые сегменты и квитанции «засоряют» сеть. Возникает вопрос: может ли протокол TCP *целенаправленно* действовать для предотвращения перегрузки? Основываясь на том, что было написано выше, читатель ответит на этот вопрос утвердительно. В самом общем виде протокол должен придерживаться следующей линии поведения:

[1] О перегрузках читайте раздел «Обратная связь для предотвращения перегрузок» главы 5, а также раздел «Поддержка QoS протоколом IP» главы 14.

Если в сети перегрузка растет, то протокол TCP на стороне отправителя уменьшает интенсивность потока сегментов, а если сеть становится менее загруженной, то отправитель увеличивает интенсивность потока сегментов.

Заметим, что в данном случае поведение протокола TCP диктуется не стремлением согласовать скорости отправителя и получателя (управление потоком), а стремлением повлиять на сетевой трафик (управление перегрузками).

Теперь нужно ответить еще на два вопроса:

- как протокол TCP узнает, когда сеть перегружена или недогружена;
- как протокол TCP регулирует интенсивность потока отправляемых сегментов.

И в этом случае ответы достаточно очевидны. Когда конечный узел не получает квитанцию на отправленный сегмент, хотя отведенный на это интервал времени уже истек, это означает, что либо сегмент данных, либо квитанция потеряны. Возможно, это произошло из-за переполнения буфера приемника, если источник отправлял данные в сеть со слишком высокой скоростью; но возможно также, что сегмент или квитанция были задержаны или вовсе потеряны из-за того, что в сети скопилось слишком много пакетов, которые ожидают в очередях перегруженных маршрутизаторов.

Если отправитель не получает вовремя квитанцию на отправленный сегмент несколько раз, то можно предположить, что сеть перегружена, напротив, если очередная ожидаемая квитанция приходит вовремя, значит, сеть работает нормально, перегрузки нет.

С помощью этих признаков протокол TCP диагностирует сеть.

Если таймер истек, а квитанция не пришла, то у отправителя есть два варианта действий: как можно быстрее отправить дубликат сегмента данных с тем, чтобы восстановить скорость потока в своем логическом соединении, либо, напротив, притормозить отправку, чтобы внести свой вклад в разгрузку сети. В первом случае, продолжая повторно передавать пакеты, протокол TCP действует исходя из «собственных» интересов. При этом он только усугубляет перегрузку сети, нанося вред в том числе и себе. Во втором случае TCP преследует «коллективное благо»: если в каждой TCP-сессии при каждой недошедшей вовремя квитанции узел-отправитель будет аналогичным образом замедлять отправку данных, это может снизить перегрузку.

При *управлении потоком* интенсивность отправки сегментов регулируется принимающей стороной, которая, исходя из заполненности своего буфера, посылает отправителю увеличенное (если свободного места много) или уменьшенное (если места мало) *окно приема*. Однако в случае управления перегрузкой решение по изменению интенсивности потока посылаемых в сеть данных должен принимать отправитель, причем не по командам от получателя, а по собственной инициативе в зависимости от загрузки сети. Для этого в протоколе TCP на стороне отправителя предусмотрен важный параметр — **окно перегрузки** (congestion window).

Процесс отправки, таким образом, регулируется двумя окнами: *окном приема*, диктуемого приемной стороной, и *окном перегрузки*, определяемым «на месте» отправителем. В качестве действующего размера окна выбирается минимальное из двух чисел: min {размер окна перегрузки, размер окна приема}.

Приведенное здесь описание работы TCP по предотвращению перегрузки все еще имеет слишком общий вид. Остаются вопросы: каким должно быть окно перегрузки в самом на-

чале работы логического соединения, насколько резким должно быть снижение скорости при обнаружении перегрузки и насколько быстро следует восстанавливать темп отправки данных в сеть, и др. Существует много реализаций протокола TCP, отличающихся тем, как их разработчики отвечают на эти вопросы. Мы сейчас кратко рассмотрим известный алгоритм управления перегрузками TCP, называемый **«Медленным Стартом»** (Slow Start), который был разработан Van Jacobson в 1988 году, стандартизован в нескольких RFC [2581, 5681 и др.] и реализован в классических версиях TCP.

Алгоритм «Медленный Старт»

На рис. 15.21 показано, как в соответствии с алгоритмом «Медленный Старт» меняется окно перегрузки.

Когда логическое соединение TCP только что установлено, протокол не имеет никакой, даже косвенной, информации о степени загруженности сети. Чтобы не терять много сегментов с самого начала (и не засорять тем самым, возможно, перегруженную сеть), в данном алгоритме выбирается осторожная тактика постепенного наращивания (медленный старт) интенсивности отправки сегментов[1]. Для этого начальный размер окна перегрузки выбирается минимальным — равным 1 сегменту.



Рис. 15.21. Пример изменения окна перегрузки

Если за время оборота (на рисунке — RTT=1) сегмент был доставлен получателю, а подтверждающая квитанция — отправителю, то, значит, сеть работает нормально, и окно перегрузки увеличивается на 1 сегмент, становясь равным 2 сегментам. Если спустя время следующего оборота (RTT=2) отправитель получит квитанции на оба отправленных сегмента, то окно перегрузки опять увеличится в два раза и станет равным 4 сегментам. Такое удвоение окна будет продолжаться до тех пор, пока либо размер окна перегрузки не сравняется с окном приема, либо до момента, когда по каким-то причинам квитанция не придет вовремя. Это событие интерпретируется как сбой (перегрузка) сети и служит «спусковым

[1] В некоторых версиях алгоритма протокол оперирует не сегментами, а байтами.

крючком» для переключения процесса с увеличения окна на его резкое уменьшение — до начального значения, равного 1.

Значение окна перегрузки *в момент сбоя* сети используется для вычисления **порогового значения**, равного половине окна перегрузки (в примере на графике максимальное значение окна перегрузки равно 16, а пороговое значение — 8).

Следующий цикл наращивания окна перегрузки происходит аналогично, но только до момента достижения им порогового значения. После этого окно перегрузки перестает удваиваться в конце каждого интервала RTT, вместо этого скорость наращивания снижается до 1 сегмента за период RTT, пока снова не произойдет событие, указывающее на перегрузку в сети. Тогда повторяется та же самая процедура — вычисление нового порогового значения, резкое снижение скорости передачи, а затем относительно медленное восстановление. Первой версией протокола TCP, включившей в себя алгоритм медленного старта, стала так называемая версия **TCP Tahoe**.

Существуют и другие стандартизованные варианты диагностики перегрузки, основанные на анализе квитанций и тайм-аутов. Например, рассмотрим ситуацию, когда N-й сегмент отправлен, установленный при его отправке таймер еще не истек, квитанция на данный сегмент еще не поступила, зато поступило 3 дубликата квитанции на предшествующий (N–1)-й, ранее уже подтвержденный сегмент. Можно сделать следующее умозаключение: поскольку дубликаты квитанции проходят по сети без задержки, значит, сеть *не перегружена* и, значит, нет смысла замедлять процесс отправки сегментов. Напротив, надо, не дожидаясь истечения таймера, быстро выполнить повторную передачу неподтвержденного сегмента [RFC5681].

Явное уведомление о перегрузке

Поскольку протокол TCP работает только на конечных узлах, он не имеет никаких средств диагностирования перегрузки в сети, кроме косвенных признаков — задержек квитанций. Иное дело маршрутизаторы. В их распоряжении данные об очередях пакетов, задержках, интенсивностях потоков и другая информация, позволяющая не только оценить текущее состояние трафика, но и предсказать возможную перегрузку в ближайшее время. Если бы протокол IP мог сообщить протоколу TCP о состоянии сети, то последний мог бы включить свои возможности по регулированию интенсивности отправки пакетов и внес бы свой вклад в оптимизацию загрузки сети.

Такая возможность, основанная на использовании двух битов поля «Тип обслуживания» в заголовке IP-пакета, реализована в методе **«Явное уведомление о перегрузке»** (Explicit Congestion Notification, ECN) [RFC 3168].

Когда маршрутизатор диагностирует предперегрузочное состояние сети, он маркирует пакет IP, содержащий сегмент TCP, устанавливая в единицу соответствующие биты заголовка IP-пакета (рис. 15.22). Маркированный пакет передается протоколом IP по адресу назначения принимающей стороне протокола TCP. Он посылает сообщение о перегрузке отправляющему протоколу TCP, который в ответ на это сообщение уменьшает вдвое окно перегрузки, как если бы он реагировал на отсутствие квитанции. В другом варианте этого метода маршрутизатор прямо направляет маркированный пакет в адрес отправителя, который отвечает на него, уменьшая окно перегрузки.

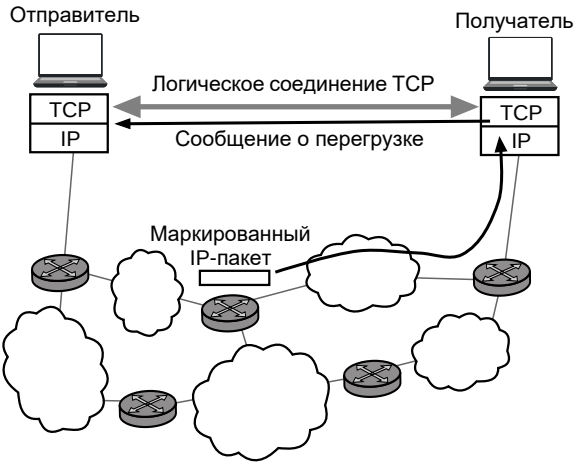


Рис. 15.22. Явное уведомление о перегрузке

Развитие протокола TCP

Транспортный уровень стека TCP/IP предлагает набор услуг вышележащему прикладному уровню. Этот набор не отличается большим разнообразием: либо быстрая, но ненадежная доставка (UDP), либо надежная, но более медленная (TCP). Разнообразие приложений приводит к разнообразию требований к услугам доставки. В одних случаях требования удовлетворяются внесением усовершенствований в классическую версию TCP (см. табл. 15.1), а в других — разрабатываются новые транспортные протоколы (см. описание протокола QUIC в разделе «Веб-служба» главы 19).

Протокол TCP, впервые опубликованный в 1974 году и стандартизованный в 1981 году (RFC 793), развивался на протяжении десятилетий. За полвека в него было внесено много усовершенствований, хотя, как ни удивительно, базовые концепции протокола TCP остались неизменными. В 2022 году на смену RFC 793 пришел новый стандарт RFC 9293, аккумулирующий все эти изменения. Наиболее важные расширения протокола в основном связаны с контролем перегрузки, восстановлением после потерь и оптимизацией производительности. Многие из них отражают специфические требования беспроводных сетей.

Таблица 15.1. Наиболее актуальные на сегодняшний день версии и расширения TCP

TCP SACK (RFC 2018) включен по умолчанию во все основные ОС	Позволяет сообщать, какие именно сегменты были получены (выборочное подтверждение). В беспроводных сетях сокращает время восстановления после всплесков потерь пакетов. Восстановление потерь на основе SACK является основой всех современных высокопроизводительных реализаций TCP
CUBIC (включен в Linux по умолчанию (RFC 8312)	Контроль перегрузки TCP, используемый по умолчанию в Linux, Android-телефонах, точках доступа Wi-Fi, 5G-модемах и др. Оптимизирован для сетей с высокой пропускной способностью и большим RTT

Таблица 15.1 (*окончание*)

BBR Google (RFC 9330+) **BBR v1** — широко используется (Google, YouTube)	Новейшее поколение, рассчитан на наличие узких мест в сети при минимальном RTT. Важное отличие — контроль перегрузки не основан на потере пакетов. Лучше других версий работает в беспроводных сетях Wi-Fi, LTE, 5G, в которых потеря пакетов вызвана радиопомехами, а не перегрузкой. Обеспечивает 2–4-кратное увеличение пропускной способности на протяженных линиях связи
HyStart++ (RFC 9406)	Используется с CUBIC для предотвращения потерь, вызванных первоначальными всплесками. Улучшает медленный старт, поэтому полезен для нестабильных беспроводных соединений
ECN явное уведомление о перегрузке (RFC 3168) и **L4S**	Сетевые технологии нового поколения, полезны в сетях с низкой задержкой, например 5G. ECN: Маршрутизаторы, обнаружив перегрузку, маркируют IP-пакеты. L4S: низкие потери, высокая масштабируемость, стремится уменьшить задержки до нескольких миллисекунд для: VR/AR, облачных игр, медиа в реальном времени
TCP Westwood+	Специально оптимизирован для беспроводных сетей. Используется в спутниковых системах, сетях БПЛА. Хорошо распознает потери пакетов, вызванные радиопомехами, и потери, происходящие из-за перегрузки сети
Multipath TCP MPTCP (RFC 8684)	Позволяет в рамках одного логического соединения параллельно передавать данные нескольких «подпотоков» через разные интерфейсы и маршруты. Используется в Apple iOS, Linux, некоторых VPN и сотовых сетях

ГЛАВА 16 Протоколы маршрутизации и технология SDN

Маршрутизация компьютерных сетей — это процесс поиска пути для пакетов данных от источника к получателю через взаимосвязанные сети. Мы уже не раз обращались к этой теме в предыдущих главах и, прежде чем перейти к более глубокому рассмотрению протоколов маршрутизации, подведем краткий итог нашим знаниям об этом важном аспекте работы сетей.

- В разделе «Обобщенная задача коммутации» главы 2 были сформулированы *базовые концепции маршрутизации*. Там было сказано, в чем состоит задача маршрутизации и чем она отличается от задачи продвижения пакетов, какая информация содержится в таблицах маршрутизации и что служит источником для этой информации.
- В главах 4 и 13 мы обсуждали место и функции протоколов маршрутизации в стандартных стеках протоколов OSI и TCP/IP соответственно.
- В главе 5 было показано, насколько важной является процедура прокладки маршрутов с учетом требований качества обслуживания.
- И наконец, в главе 14 подробно рассматривалась работа протокола IP по продвижению пакетов через сеть на основе таблиц маршрутизации.

В этой главе дан обзор различных подходов к маршрутизации и описание алгоритмов протоколов RIP, OSPF, BGP и протоколов группового вещания. Завершает главу раздел, посвященный программно-определяемым сетям SDN.

Обзор методов маршрутизации

Существует много разных подходов к маршрутизации, отличающихся способами нахождения и фиксации маршрутов, а также тем, какие критерии используются для выбора оптимального маршрута, в какие моменты изменяется содержание таблиц маршрутизации, насколько эффективными являются алгоритмы, положенные в основу того или иного метода маршрутизации, и т. д.

Наиболее простые методы вообще *не требуют наличия таблиц маршрутизации на маршрутизаторах*. Например, **лавинная маршрутизация** заключается в том, что каждый маршрутизатор передает пакет всем своим непосредственным соседям, исключая тот, от которого его получил. Понятно, что это не самый рациональный способ, так как пропускная способность сети используется крайне расточительно. Тем не менее такой подход работоспособен (именно так мосты и коммутаторы локальных сетей поступают с кадрами, имеющими неизвестные адреса).

Еще одним видом маршрутизации, не требующим таблиц маршрутизации, является маршрутизация от источника (source routing). В этом случае отправитель помещает в пакет последовательность адресов узлов сети, через которые должен проходить пакет, чтобы дойти до получателя. То есть маршрут помещается в заголовок пакета в его исходном виде, не разделенный на отдельные записи в таблицах каждого маршрутизатора. Маршрут может быть задан либо вручную администратором, либо автоматически узлом-отправителем, но в этом случае ему нужно поддерживать какой-либо протокол маршрутизации, который сообщит ему о топологии и состоянии сети. Маршрутизация от источника была опробована на этапе зарождения интернета и сохранилась как практически неиспользуемая возможность протокола IPv4. В IPv6 и MPLS маршрутизация от источника является одним из стандартных режимов продвижения пакетов, который поддерживается протоколом **сегментной маршрутизации** (segment routing) SRv6 и SR MPLS (читайте об этом протоколе в главе 20 в разделе «Инжиниринг трафика в MPLS»). Сегментная маршрутизация является обобщением маршрутизации от источника, когда наряду с последовательностью узлов, составляющих маршрут, для каждого промежуточного узла может задаваться инструкция, требующая от узла выполнения некоторого дополнительного действия над пакетом, например, компрессии заголовка, создания временной отметки или глубокой проверки пакета для выявления атаки злоумышленника.

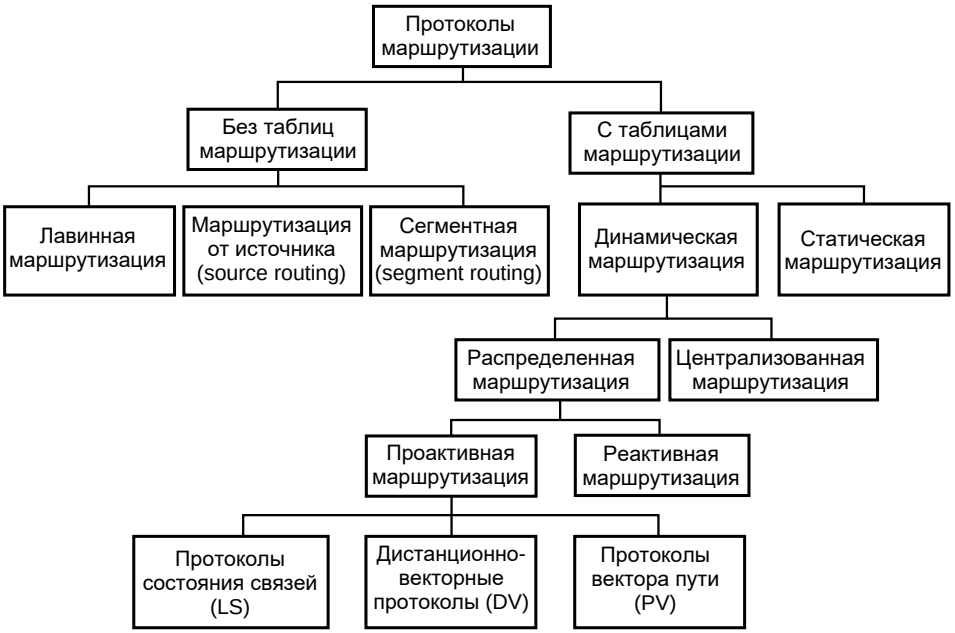


Рис. 16.1. Разнообразие подходов к маршрутизации

Большинство же протоколов маршрутизации нацелено на *создание таблиц маршрутизации*. Выбор рационального маршрута может осуществляться на основании различных *критериев*. Сегодня в IP-сетях применяются протоколы маршрутизации, в которых маршрут выбирается по критерию кратчайшего расстояния. При этом расстояние измеряется в различных **метриках**. Чаще всего используется простейшая метрика — количество хопов, то

есть количество маршрутизаторов, которые нужно преодолеть пакету до сети назначения. В качестве метрик применяются также пропускная способность и надежность каналов, вносимые ими задержки и любые комбинации этих метрик. Например, составная метрика протокола Cisco EIGRP выглядит как достаточно сложная функция от пропускной способности, задержки, нагрузки и надежности линий связи.

В некоторых случаях решения о выборе маршрута основаны не на метрике, а на политике. **Политика** — это набор правил, который может диктовать выбор маршрута для пакета не только в зависимости от адреса назначения, как это бывает обычно, и на основании критериев оптимальности маршрута, а в зависимости от адреса отправителя, типа приложения, сгенерировавшего данный пакет, административных и организационных соображений. В протоколе BGP, например, решения о пересылке данных могут приниматься на основе примерно такой политики: «Предпочитать маршруты, которые проходят через следующие (...) дружественные автономные системы, и избегать следующие (...) нежелательные автономные системы».

Задачей протоколов маршрутизации является создание на всех маршрутизаторах *согласованных* друг с другом таблиц маршрутизации, то есть таких таблиц, которые обеспечат доставку пакета от исходной сети в сеть назначения за конечное число шагов. Различают протоколы, выполняющие статическую и динамическую (адаптивную) маршрутизацию.

При статической маршрутизации все записи в таблице имеют неизменяемый, фиксированный статус, что подразумевает бесконечный срок их жизни. Записи о маршрутах составляются и вводятся в память каждого маршрутизатора вручную администратором сети. При изменении состояния сети администратору необходимо срочно отразить эти изменения в соответствующих таблицах маршрутизации, иначе может произойти их рассогласование, и сеть будет работать некорректно. Статическая маршрутизация подходит для небольших сетей и сетей доступа, а также в тех случаях, когда маршрут является критически важным с точки зрения безопасности.

При **динамической (адаптивной) маршрутизации** изменения конфигурации сети *автоматически* (а не вручную) отражаются в таблицах маршрутизации благодаря работе *протоколов маршрутизации*. Эти протоколы собирают информацию о топологии связей в сети, что позволяет им оперативно отражать все текущие изменения. В таблицах маршрутизации при адаптивной маршрутизации обычно имеется информация об интервале времени, в течение которого данный маршрут будет оставаться действительным. Это время называют *временем жизни* (TTL) маршрута. Если по истечении времени жизни существование маршрута не подтверждается протоколом маршрутизации, то он считается нерабочим и пакеты по нему больше не посылаются.

Протоколы динамической маршрутизации бывают распределенными и централизованными. При **распределенном подходе** все маршрутизаторы сети находятся в равных условиях: они находят маршруты и строят собственные таблицы маршрутизации, работая в тесной кооперации друг с другом, постоянно обмениваясь информацией о конфигурации сети. Недостатком распределенных протоколов маршрутизации является существование периодов нестабильной работы сети, вызванной временной несогласованностью таблиц разных маршрутизаторов. Распределенному протоколу маршрутизации обычно нужно некоторое время — *время конвергенции*, чтобы после нескольких итераций обмена служебной информацией все маршрутизаторы сети внесли изменения в свои таблицы, в результате чего они снова стали бы согласованными. Различные протоколы маршрутизации обладают разным временем конвергенции, так, протокол RIP характеризуется медленной сходимо-

стью, а OSPF и IS-IS — быстрой сходимостью. Особенно длительное время конвергенции у протокола BGP.

При **централизованном подходе** в сети существует один выделенный маршрутизатор (сервер маршрутов), собирающий всю информацию о топологии и состоянии сети от других маршрутизаторов. На основании этих данных выделенный маршрутизатор строит для остальных маршрутизаторов таблицы маршрутизации, распространяя их затем по сети, чтобы каждый маршрутизатор получил собственную таблицу и в дальнейшем самостоятельно принимал решение о продвижении каждого пакета. Централизованный подход порождает более рациональные маршруты, чем распределенный, так как решения принимаются на основе полной и непротиворечивой информации о топологии связей сети.

Одним из вариантов централизованного подхода к маршрутизации являются **программно-определяемые сети** (Software Defined Networks, **SDN**). В этом случае вся логика продвижения пакетов переносится в программируемый контроллер сети, а маршрутизаторы сети являются достаточно простыми устройствами, просто отрабатывающими инструкции, поступающие от контроллера сети (подход SDN рассматривается далее в этой главе). Недостатком централизованного подхода является *плохая масштабируемость*, и он более подходит для небольших сетей. Из-за этого недостатка в интернете доминировали и продолжают доминировать распределенные протоколы маршрутизации.

Применяемые сегодня в IP-сетях протоколы маршрутизации относятся к динамическим распределенным протоколам, которые в свою очередь делятся на группы:

- **Дистанционно-векторные протоколы** (Distance Vector, **DV**) — маршрутизаторы обмениваются информацией о достижимых сетях и расстояниях до них. Пример: протокол RIP.
- **Протоколы состояния связей** (Link State, **LS**) — маршрутизаторы рассылают объявления о состоянии линии связи, строят граф сети и на этом графе находят кратчайшие пути. Примеры: протоколы OSPF и IS-IS.
- **Протоколы вектора пути** (Path Vector, **PV**) — маршрутизаторы анонсируют все пути в виде перечня номеров автономных систем (Autonomous System, AS), то есть крупных административных объединений сетей. Пример: протокол BGP.

В **дистанционно-векторных протоколах** каждый маршрутизатор *периодически* и *широковещательно* рассылает по сети вектор, компонентами которого являются расстояния (дистанции), измеренные в той или иной метрике, от данного маршрутизатора до всех известных ему сетей.

Дистанционно-векторные протоколы используют для нахождения оптимального пути алгоритм Беллмана — Форда. Он основан на рассылке каждым узлом своим соседям по сети вектора расстояний (дистанций) от него до всех известных ему узлов. Получая такую информацию от соседей, узел выбирает маршрут с наименьшим суммарным расстоянием до некоторого узла назначения. Теорема Беллмана — Форда доказывает, что за конечное число итераций (получение новых объявлений от соседних узлов) каждый узел находит оптимальные маршруты до всех остальных узлов сети.

Пакеты дистанционно-векторных протоколов маршрутизации обычно называют **объявлениями о расстояниях**, так как с их помощью маршрутизатор объявляет остальным маршрутизаторам известные ему сведения о расстояниях от него до известных ему узлов сети.

Дистанционно-векторные алгоритмы хорошо работают только в небольших сетях. В больших же они периодически засоряют линии связи интенсивным трафиком. К тому же изменения конфигурации не всегда корректно могут отражаться алгоритмом этого типа, так как маршрутизаторы не имеют точного представления о топологии связей в сети, а располагают только косвенной информацией — вектором расстояний. Наиболее распространенным протоколом, основанным на дистанционно-векторном алгоритме, является протокол RIP.

Протоколы состояния связей обеспечивают каждый маршрутизатор информацией, достаточной для построения *точного* графа связей всей сети в целом. Все маршрутизаторы работают на основании одного и того же алгоритма маршрутизации и на основе одного и того же графа сети, что делает процесс маршрутизации более устойчивым к изменениям конфигурации. Чтобы понять, в каком состоянии находятся линии связи, подключенные к его портам, маршрутизатор периодически обменивается короткими пакетами HELLO со своими непосредственными соседями. В отличие от протоколов DV, которые регулярно передают вектор расстояний, протоколы LS ограничиваются короткими сообщениями, а передача более объемных сообщений происходит только в тех случаях, когда с помощью сообщений HELLO был установлен факт изменения состояния какой-либо связи.

На основе сообщений о состоянии связей каждый узел сети строит граф сети, при этом графы сети у всех узлов идентичны. Затем каждый узел находит на графе сети кратчайшие пути до всех остальных узлов и запоминает маршруты к ним. Поиск оптимальных путей на графе может выполняться различными алгоритмами. Наиболее часто используется интеграционный алгоритм Дейкстры.

Алгоритм Дейкстры — это так называемый «жадный» способ поиска кратчайших путей: на каждой итерации он всегда делает локально лучший (в определенном смысле) выбор до некоторой соседней вершины, и в конце концов после конечного числа итераций этот выбор оказывается глобально лучшим.

Служебный трафик, создаваемый протоколами LS, гораздо менее интенсивный, чем у протоколов DV. Протоколами, основанными на алгоритме состояния связей, являются протокол IS-IS стека OSI (этот протокол используется также в стеке TCP/IP) и протокол OSPF стека TCP/IP.

Еще одна концепция маршрутизации связана с тем, в какие моменты должна обновляться информация в таблицах маршрутизации. В этом отношении динамические протоколы делятся на проактивные и реактивные.

Проактивная (или таблично-управляемая) маршрутизация — это постоянное обновление таблиц маршрутизации при каждом изменении конфигурации сети. Почти все протоколы маршрутизации (OSPF, IS-IS, BGP, RIP, EIGRP и т. д.) *проводных* сетей являются проактивными, потому что изменения топологии в таких сетях происходят относительно редко и пропускная способность достаточна для того, чтобы трафик протокола маршрутизации не слишком загружал сеть. При проактивной маршрутизации таблицы маршрутизации обновляются вне зависимости от трафика в сети и запись о некоторой сети попадает в таблицу маршрутизации независимо от того, направляются ли в эту сеть пакеты или нет.

Реактивная маршрутизация предполагает нахождение маршрутов только когда в них есть необходимость, то есть тогда, когда требуется отправить данные. Такой подход впервые появился в мобильных беспроводных сетях, для которых характерны настолько интенсивные изменения топологии, что задача постоянного отслеживания актуальных маршрутов становится слишком ресурсоемкой (вычислительные затраты маршрутизаторов,

пропускная способность линий связи). Поэтому выбор между вариантами «всегда поддерживать маршруты» (проактивный подход) или «находить маршруты по требованию» (реактивный подход) стал часто решаться в мобильных беспроводных сетях в пользу последнего. Примеры реактивных протоколов приведены в разделе «Маршрутизация в сетях дронов» главы 27.

Во всех рассмотренных выше случаях мы имели дело с **одноадресными** протоколами, которые находят маршруты для связей типа «один к одному». В задачах группового вещания, которые рассматриваются далее в этой главе, мы встречаем протоколы **многоадресной** (групповой) маршрутизации, используемые для связи «один ко многим» или «многие ко многим». Здесь протокол маршрутизации должен находить пути доставки сообщения, адресованного сразу нескольким получателям. Изначально отправитель генерирует только один экземпляр сообщения с *групповым адресом (multicast)*. Протоколы многоадресной маршрутизации **строят** граф (покрывающее дерево), связывающий все хосты в определенной группе. Сообщение перемещается вдоль графа и копируется на каждой из «развилок», ведущих к тому или иному члену группы, указанной в адресе данного сообщения. В конце концов, пакет с групповым адресом достигает маршрутизатора, к которому *непосредственно* подключена сеть с хостами — членами данной группы.

Подходы к маршрутизации могут также отличаться количеством предлагаемых путей между отправителем и получателем. Обычно это один маршрут, но в некоторых случаях для поддержки баланса загрузки или обеспечения повышенной отказоустойчивости используются несколько маршрутов. Например, сегментная маршрутизация MPLS предлагает несколько маршрутов, называя их путями-кандидатами. Эти пути отличаются алгоритмом, их создавшим: один путь может быть создан алгоритмом кратчайшего пути, другой — алгоритмом инжиниринга трафика со специфическими требованиями к маршруту, третий — вручную администратором сети.

Отметим еще один важный признак классификации протоколов маршрутизации, разделяющий все протоколы маршрутизации на два очень неравных класса: **протоколы внешнего шлюза** (Exterior Gateway Protocol, **EGP**), работающие между автономными системами, к которым в настоящее время относится только один протокол BGP, и **протоколы внутреннего шлюза** (Interior Gateway Protocol, **IGP**), работающие в пределах одной автономной системы, к которым относятся практически все рассмотренные ранее протоколы. Подробнее о протоколах EGP и IGP читайте далее в этой главе.

Заметим в заключение, что существует немало протоколов маршрутизации, которые сложно отнести к одному определенному классу, потому что они объединяют в себе свойства разных классов. Такие протоколы называют **гибридными**. Например, протокол сегментной маршрутизации SRv6 обладает свойствами не только маршрутизации от источника, но и состояния связей, а протокол DSDV, предназначенный для AdHoc-сетей, сочетает характерные особенности дистанционно-векторного протокола DV и протокола состояния связей LS.

Протокол RIP

Протокол маршрутной информации RIP (Routing Information Protocol) является протоколом маршрутизации дистанционно-векторного типа и чаще всего используется в небольших сетях. Для IP-сетей имеются две версии RIP: RIPv1 (не поддерживает ма-

ски) и RIPv2 (поддерживает маски). Так как способ построения таблиц маршрутизации в обеих версиях протокола принципиально не отличается, мы ограничимся описанием работы версии 1.

Построение таблицы маршрутизации

Для измерения расстояния до сети стандарты протокола RIP допускают различные метрики: хопы, значения пропускной способности, вносимые задержки, надежность сетей (то есть соответствующие признакам D, T и R в поле качества сервиса IP-пакета), а также любые комбинации этих метрик. Метрика должна обладать свойством *аддитивности:* метрика составного пути должна быть равна сумме метрик составляющих отрезков этого пути. В большинстве реализаций RIP используется простейшая метрика — количество хопов, то есть количество промежуточных маршрутизаторов, которые нужно преодолеть пакету до сети назначения. Рассмотрим процесс построения таблицы маршрутизации с помощью протокола RIP на примере составной сети (рис. 16.2), разделив этот процесс на 5 этапов.

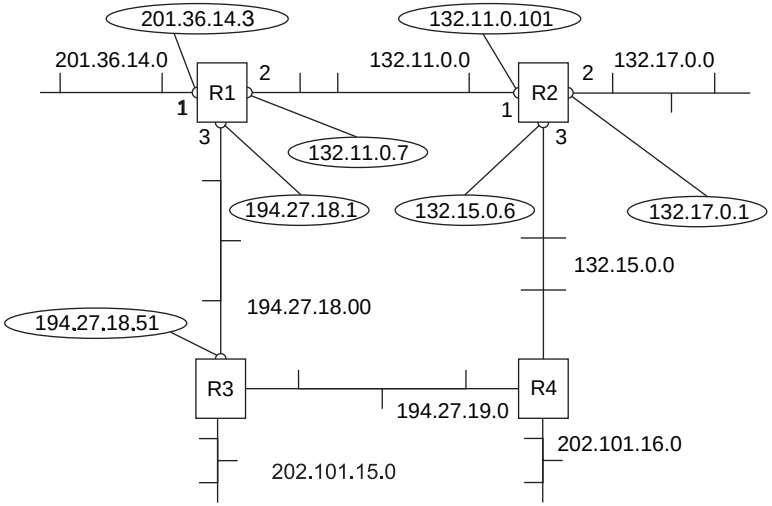


Рис. 16.1. Сеть, построенная на маршрутизаторах RIP

Этап 1 — создание минимальной таблицы. Данная составная сеть включает восемь IP-сетей, связанных четырьмя маршрутизаторами с идентификаторами: R1, R2, R3 и R4. Маршрутизаторы, работающие по протоколу RIP, могут иметь идентификаторы, однако для протокола они не являются необходимыми. В RIP-сообщениях эти идентификаторы не передаются. В исходном состоянии на каждом маршрутизаторе программным обеспечением стека TCP/IP автоматически создается минимальная таблица маршрутизации, в которой учитываются только непосредственно подсоединенные сети. На рисунке адреса портов маршрутизаторов (в отличие от адресов сетей) помещены в овалы. Таблица 16.1 позволяет оценить примерный вид минимальной таблицы маршрутизации маршрутизатора R1 (минимальные таблицы маршрутизации в других маршрутизаторах выглядят аналогично).

Таблица 16.1. Минимальная таблица маршрутизации маршрутизатора R1

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	201.36.14.3	1	1
132.11.0.0	132.11.0.7	2	1
194.27.18.0	194.27.18.1	3	1

Этап 2 — рассылка минимальной таблицы соседям. После инициализации каждый маршрутизатор начинает посылать своим соседям сообщения протокола RIP, в которых содержится информация из его минимальной таблицы. RIP-сообщения передаются в дейтаграммах протокола UDP и включают два параметра для каждой сети: ее IP-адрес и расстояние до нее от передающего сообщение маршрутизатора.

По отношению к любому маршрутизатору соседями являются те маршрутизаторы, которым данный маршрутизатор может передать IP-пакет по какой-либо своей сети, не пользуясь услугами промежуточных маршрутизаторов. Например, для маршрутизатора R1 соседями являются маршрутизаторы R2 и R3, а для маршрутизатора R4 — маршрутизаторы R2 и R3. Таким образом, маршрутизатор R1 передает маршрутизаторам R2 и R3 следующие сообщения:

- сеть 201.36.14.0, расстояние 1;
- сеть 132.11.0.0, расстояние 1;
- сеть 194.27.18.0, расстояние 1.

Этап 3 — получение RIP-сообщений от соседей и обработка полученной информации. После получения аналогичных сообщений от маршрутизаторов R2 и R3 маршрутизатор R1 наращивает каждое полученное поле метрики на единицу и запоминает, через какой порт и от какого маршрутизатора получена новая информация (адрес этого маршрутизатора станет адресом следующего маршрутизатора, если эта запись будет внесена в таблицу маршрутизации). Затем маршрутизатор начинает сравнивать новую информацию с той, которая хранится в его таблице маршрутизации (табл. 16.2).

Таблица 16.2. Таблица маршрутизации маршрутизатора R1

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	201.36.14.3	1	1
132.11.0.0	132.11.0.7	2	1
194.27.18.0	194.27.18.1	3	1
132.17.0.0	132.11.0.101	2	2
132.15.0.0	132.11.0.101	2	2
194.27.19.0	194.27.18.51	3	2
202.101.15.0	194.27.18.51	3	2
~~132.11.0.0~~	~~132.11.0.101~~	~~2~~	~~2~~
~~194.27.18.0~~	~~194.27.18.51~~	~~3~~	~~2~~

Записи с четвертой по девятую получены от соседних маршрутизаторов, все они претендуют на помещение в таблицу. Однако только записи с четвертой по седьмую попадают в таблицу, а записи восьмая и девятая — нет. Это происходит потому, что они содержат данные об уже имеющихся в таблице маршрутизатора R1 сетях, а расстояние до них больше, чем в существующих записях. Протокол RIP замещает запись о какой-либо сети только в том случае, если новая информация имеет лучшую метрику (с меньшим расстоянием в хопах), чем имеющаяся. В результате в таблице маршрутизации о каждой сети остается только одна запись; если же имеется несколько записей, равнозначных в отношении путей к одной и той же сети, то все равно в таблице остается одна запись, которая пришла в маршрутизатор первой по времени. Для этого правила существует исключение — если худшая информация о какой-либо сети пришла от того же маршрутизатора, на основании сообщения которого была создана данная запись, то худшая информация замещает лучшую.

Этап 4 — рассылка новой таблицы соседям. Каждый маршрутизатор отсылает новое RIP-сообщение всем своим соседям. В этом сообщении он помещает данные обо всех известных ему сетях — как непосредственно подключенных, так и удаленных, о которых маршрутизатор узнал из RIP-сообщений.

Этап 5 — получение RIP-сообщений от соседей и обработка полученной информации. Этап 5 повторяет этап 3 — маршрутизаторы принимают RIP-сообщения, обрабатывают содержащуюся в них информацию и на ее основании корректируют свои таблицы маршрутизации. Посмотрим, как это делает маршрутизатор R1 (табл. 16.3).

Таблица 16.3. Таблица маршрутизации маршрутизатора R1

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	201.36.14.3	1	1
132.11.0.0	132.11.0.7	2	1
194.27.18.0	194.27.18.1	3	1
132.17.0.0	132.11.0.101	2	2
132.15.0.0	132.11.0.101	2	2
~~132.15.0.0~~	~~194.27.18.51~~	~~3~~	~~3~~
194.27.19.0	194.27.18.51	3	2
~~194.27.19.0~~	~~132.11.0.101~~	~~2~~	~~3~~
~~202.101.15.0~~	~~194.27.18.51~~	~~3~~	~~2~~
202.101.16.0	132.11.0.101	2	3
~~202.101.16.0~~	~~194.27.18.51~~	~~3~~	~~3~~

На этом этапе маршрутизатор R1 получает от маршрутизатора R3 информацию о сети 132.15.0.0, которую тот, в свою очередь, на предыдущем цикле работы получил от маршрутизатора R4. Маршрутизатор уже знает о сети 132.15.0.0, причем старая информация имеет лучшую метрику, чем новая, поэтому новая информация об этой сети отбрасывается. О сети 202.101.16.0 маршрутизатор R1 узнает на этом этапе впервые, причем данные о ней приходят от двух соседей — от R3 и R4. Поскольку метрики в этих сообщениях указаны одинаковые, в таблицу попадают данные, пришедшие первыми. В нашем примере счита-

ется, что маршрутизатор R2 опередил маршрутизатор R3 и первым переслал свое RIP-сообщение маршрутизатору R1.

Если маршрутизаторы периодически повторяют этапы рассылки и обработки RIP-сообщений, то за конечное время в сети установится *корректный режим маршрутизации*. Под корректным режимом маршрутизации здесь понимается такое состояние таблиц маршрутизации, когда все сети достижимы из любой сети с помощью некоторого рационального маршрута. Пакеты будут доходить до адресатов и не зацикливаться в петлях, подобных той, которая образуется на рис. 16.2 маршрутизаторами R1, R2, R3 и R4. Если в сети все маршрутизаторы, их интерфейсы и соединяющие их линии связи остаются работоспособными, то объявления по протоколу RIP можно делать достаточно редко, например один раз в день. Однако изменения в сетях постоянно происходят — меняется работоспособность маршрутизаторов и линий связи, кроме того, маршрутизаторы и линии связи могут добавляться в существующую сеть или же выводиться из ее состава. Для адаптации к изменениям в сети протокол RIP использует ряд механизмов.

Адаптация маршрутизаторов RIP к изменениям состояния сети

К новым маршрутам маршрутизаторы RIP приспосабливаются просто — они передают новую информацию в очередном сообщении своим соседям, и постепенно эта информация становится известна всем маршрутизаторам сети. К изменениям, связанным с потерей какого-либо маршрута, маршрутизаторы RIP адаптируются сложнее, поскольку в формате сообщений протокола RIP нет поля, которое бы указывало на то, что путь к данной сети больше не существует. Для уведомления о том, что некоторый маршрут недействителен, используются два механизма:

- истечение времени жизни маршрута;
- указание специального (бесконечного) расстояния до сети, ставшей недоступной.

Механизм *истечения времени жизни маршрута* основан на том, что каждая запись таблицы маршрутизации (как и записи таблицы продвижения моста/коммутатора), полученная по протоколу RIP, имеет время жизни (TTL). При поступлении очередного RIP-сообщения, которое подтверждает справедливость данной записи, таймер времени жизни устанавливается в исходное состояние, а затем из него каждую секунду вычитается единица. Если за время тайм-аута не придет новое сообщение об этом маршруте, он помечается как *недействительный*.

Значение тайм-аута связано с периодом рассылки векторов по сети. В протоколе RIP период рассылки выбран равным 30 секундам, а тайм-аут — шестикратному значению периода рассылки, то есть 180 секундам. Шестикратный запас времени нужен для уверенности в том, что сеть действительно стала недоступной, а не просто произошли потери RIP-сообщений (а это возможно, так как протокол RIP использует транспортный протокол UDP, который не обеспечивает надежной доставки сообщений). Если какой-либо маршрутизатор перестает корректно работать и слать своим соседям сообщения о достижимых через него сетях, то через 180 секунд все записи, порожденные этим маршрутизатором, у его ближайших соседей станут недействительными. После этого процесс повторится уже для соседей ближайших соседей — они вычеркнут подобные записи лишь через 360 секунд.

Как видно, сведения о сетях, пути к которым не могут проходить через отказавший маршрутизатор, распространяются по сети не очень быстро. В этом заключается одна из причин выбора в качестве периода рассылки небольшой величины в 30 секунд. Механизм таймаута работает в тех случаях, когда маршрутизатор не может послать соседям сообщение об отказавшем маршруте, так как либо сам неработоспособен, либо неработоспособна линия связи, по которой можно было бы передать сообщение.

Когда же сообщение послать можно, маршрутизаторы RIP используют прием, заключающийся в *указании бесконечного расстояния до сети, ставшей недоступной.* В протоколе RIP бесконечным условно считается расстояние в 16 хопов. Получив сообщение, в котором расстояние до некоторой сети равно 16, маршрутизатор должен проверить, исходит ли эта «плохая» информация о сети от того же маршрутизатора, сообщение которого послужило в свое время основанием для записи о данной сети в таблице маршрутизации. Если это тот же маршрутизатор, то информация считается достоверной и маршрут помечается как недоступный.

Причиной выбора в качестве «бесконечного» расстояния столь небольшого числа является то, что в некоторых случаях отказы связей в сети вызывают длительные периоды некорректной работы маршрутизаторов RIP, выражающейся в зацикливании пакетов в петлях сети. И чем меньше расстояние, используемое в качестве «бесконечного», тем такие периоды короче.

Пример зацикливания пакетов

Рассмотрим случай зацикливания пакетов на примере сети, изображенной на рис. 16.2. Пусть маршрутизатор R1 обнаружил, что его связь с непосредственно подключенной сетью 201.36.14.0 потеряна (например, по причине отказа интерфейса 201.36.14.3). Маршрутизатор R1 отмечает в своей таблице маршрутизации, что сеть 201.36.14.0 недоступна. В худшем случае он обнаружит это сразу же после отправки очередных RIP-сообщений, так что до начала нового цикла его объявлений, в котором он должен сообщить соседям, что расстояние до сети 201.36.14.0 стало равным 16, остается почти 30 секунд. Каждый маршрутизатор работает на основании своего внутреннего таймера, не синхронизируя работу по рассылке объявлений с другими маршрутизаторами. Поэтому весьма вероятно, что маршрутизатор R2 опередит маршрутизатор R1 и передаст ему свое сообщение раньше, чем R1 успеет передать новость о недостижимости сети 201.36.14.0. А в этом сообщении имеются данные, порожденные записью в таблице маршрутизации R2 (табл. 16.4).

Таблица 16.4. Таблица маршрутизации маршрутизатора R2

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	132.11.0.7	1	2

Эта запись, полученная от маршрутизатора R1, была корректна до отказа интерфейса 201.36.14.3, но теперь она устарела, причем маршрутизатор R2 об этом не знает. Далее маршрутизатор R1 получает новую информацию о сети 201.36.14.0 — эта сеть достижима через маршрутизатор R2 с метрикой 2. Раньше маршрутизатор R1 также получал эту информацию от R2, но игнорировал ее, так как его собственная метрика для 201.36.14.0 была

лучше. Теперь R1 должен принять данные о сети 201.36.14.0, полученные от R2, и заменить запись в таблице маршрутизации о недостижимости этой сети (табл. 16.5). *В результате в сети образуется маршрутная петля*: пакеты, направляемые узлам сети 201.36.14.0, станут передаваться маршрутизатором R2 маршрутизатору R1, а маршрутизатор R1 будет возвращать их маршрутизатору R2.

Таблица 16.5. Таблица маршрутизации маршрутизатора R1

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	132.11.0.101	2	3

IP-пакеты продолжат циркулировать по этой петле до тех пор, пока не истечет время жизни каждого пакета. Рассмотрим периоды времени, кратные времени жизни записей в таблицах маршрутизаторов.

- *Время 0–180 с.* После отказа интерфейса в маршрутизаторах R1 и R2 начинают сохраняться некорректные записи. Маршрутизатор R2 по-прежнему снабжает маршрутизатор R1 своей записью о сети 201.36.14.0 с метрикой 2, так как ее время жизни не истекло. Пакеты зацикливаются.
- *Время 180–360 с.* В начале этого периода у маршрутизатора R2 истекает время жизни записи о сети 201.36.14.0 с метрикой 2, так как маршрутизатор R1 в предыдущий период посылал ему сообщения о сети 201.36.14.0 с худшей метрикой, чем у R2, и они не могли подтвердить эту запись. Теперь маршрутизатор R2 принимает от маршрутизатора R1 запись о сети 201.36.14.0 с метрикой 3 и трансформирует ее в запись с метрикой 4. Маршрутизатор R1 не получает новых сообщений от маршрутизатора R2 о сети 201.36.14.0 с метрикой 2, поэтому время жизни его записи начинает уменьшаться. Пакеты продолжают зацикливаться.
- *Время 360–540 с.* У маршрутизатора R1 истекает время жизни записи о сети 201.36.14.0 с метрикой 3. Маршрутизаторы R1 и R2 опять меняются ролями — R2 снабжает R1 устаревшей информацией о пути к сети 201.36.14.0, уже с метрикой 4, которую R1 преобразует в метрику 5. Пакеты продолжают зацикливаться.

Если бы в протоколе RIP не было выбрано расстояние 16 в качестве недостижимого, то описанный процесс длился бы бесконечно (вернее, пока не была бы исчерпана разрядная сетка поля расстояния), поэтому эта проблема получила название «счет до бесконечности». В результате маршрутизатор R2 на очередном этапе описанного процесса получает от маршрутизатора R1 метрику 15, которая после наращивания, превращаясь в метрику 16, фиксирует недостижимость сети. Таким образом, в нашем примере период нестабильной работы сети длится 36 минут!

Ограничение в 15 хопов сужает область применения протокола RIP до сетей, в которых число промежуточных маршрутизаторов не превышает 15. Для более масштабных сетей нужно применять другие протоколы маршрутизации, например OSPF, или разбивать сеть на автономные области. Приведенный пример хорошо иллюстрирует главную причину нестабильности маршрутизаторов, работающих по протоколу RIP. Эта причина коренится в самом принципе работы дистанционно-векторных протоколов — использовании информации, полученной из «вторых рук». Действительно, маршрутизатор R2 передает маршрутизатору R1 информацию о достижимости сети 201.36.14.0, за достоверность которой он сам не отвечает.

ПРИМЕЧАНИЕ

Не следует думать, что при любых отказах интерфейсов и маршрутизаторов в сетях возникают маршрутные петли. Если бы маршрутизатор R1 успел передать сообщение о недостижимости сети 201.36.14.0 раньше ложной информации маршрутизатора R2, то маршрутная петля не образовалась бы. Так что маршрутные петли даже без дополнительных методов борьбы с ними возникают в среднем не более чем в половине потенциально возможных случаев.

Методы борьбы с ложными маршрутами в протоколе RIP

Хотя протокол RIP не в состоянии полностью исключить в сети переходные состояния, когда некоторые маршрутизаторы пользуются устаревшей информацией о несуществующих маршрутах, имеется несколько методов, позволяющих во многих случаях решать подобные проблемы.

Проблема с петлей, образующейся между соседними маршрутизаторами, надежно решается с помощью метода **расщепления горизонта**, заключающегося в том, что маршрутная информация о некоторой сети, хранящаяся в таблице маршрутизации, никогда не передается тому маршрутизатору, от которого она получена.

Практически все сегодняшние маршрутизаторы, работающие по протоколу RIP, используют технику расщепления горизонта. Если бы маршрутизатор R2 в рассмотренном ранее примере поддерживал технику расщепления горизонта, то он бы не передал маршрутизатору R1 устаревшую информацию о сети 201.36.14.0, так как получил он ее именно от маршрутизатора R1. Однако расщепление горизонта не помогает, если петли образуются не двумя, а большим числом маршрутизаторов.

Рассмотрим более детально ситуацию, которая возникнет в сети, приведенной на рис. 16.2, в случае потери связи маршрутизатора R1 с сетью 201.36.14.0. Пусть все маршрутизаторы этой сети поддерживают технику расщепления горизонта. В этой ситуации маршрутизаторы R2 и R3 не возвращают маршрутизатору данные о сети 201.36.14.0 с метрикой 2, так как они получили эту информацию от маршрутизатора R1. Однако они передают маршрутизатору информацию о достижимости сети 201.36.14.0 с метрикой 4 через себя, так как получили эту информацию по сложному маршруту, а не непосредственно от маршрутизатора R1. Например, маршрутизатор R2 получает эту информацию по цепочке R4-R3-R1, поэтому маршрутизатор R1 снова может быть обманут, пока каждый из маршрутизаторов в цепочке R3-R4-R2 не вычеркнет запись о достижимости сети 201.36.14.0. Для предотвращения зацикливания пакетов по составным петлям при отказах связей применяются два других приема: триггерные обновления и замораживание изменений.

Прием **триггерных обновлений** состоит в том, что маршрутизатор, получив данные об изменении метрики до какой-либо сети, не ждет истечения периода передачи таблицы маршрутизации, а передает данные об изменившемся маршруте немедленно. Этот прием может во многих случаях предотвратить передачу устаревших сведений об отказавшем маршруте, но он перегружает сеть служебными сообщениями, поэтому триггерные объявления также делаются с некоторой задержкой. По этой причине возможна ситуация, когда регулярное обновление в каком-либо маршрутизаторе чуть опережает по времени приход триггерного

обновления от предыдущего в цепочке маршрутизатора, и данный маршрутизатор успевает передать по сети устаревшую информацию о несуществующем маршруте.

Второй прием — **замораживание изменений** — позволяет исключить подобные ситуации. Он связан с введением тайм-аута на принятие новых данных о сети, которая только что стала недоступной. Этот тайм-аут предотвращает принятие устаревших сведений о некотором маршруте от тех маршрутизаторов, которые находятся на некотором расстоянии от отказавшей связи и передают устаревшие сведения о ее работоспособности. Предполагается, что в течение тайм-аута замораживания изменений эти маршрутизаторы вычеркнут данный маршрут из своих таблиц, так как не получат о нем новых записей и не будут распространять устаревшие сведения по сети.

Протокол OSPF

Протокол **OSPF** (Open Shortest Path First — выбор кратчайшего пути первым) основан на алгоритме состояния связей и обладает многими свойствами, способствующими его применению в больших гетерогенных сетях, в первую очередь — представление большой сети в виде нескольких связанных областей и применение двухуровневой маршрутизации, внутри области и между областями.

Два этапа построения таблицы маршрутизации

OSPF разбивает процедуру построения таблицы маршрутизации на два этапа:

Этап 1 — построение и поддержание базы данных о состоянии связей сети. Связи сети могут быть представлены в виде графа, в котором вершинами являются маршрутизаторы и подсети, а ребрами — связи между ними. Каждый маршрутизатор обменивается со своими соседями той информацией о графе сети, которой он располагает к данному моменту. Процесс похож на процесс распространения векторов расстояний до сетей в протоколе RIP, но сама информация качественно иная — это информация о *топологии* сети. Сообщения, с помощью которых распространяется топологическая информация, называются **объявлениями о состоянии связей** (Link State Advertisements, **LSA**) сети.

При транзитной передаче объявлений LSA маршрутизаторы не модифицируют информацию, как это происходит в дистанционно-векторных протоколах, в частности в RIP, а передают ее в неизменном виде. В результате все маршрутизаторы сети сохраняют в своей памяти идентичные сведения о текущей конфигурации графа связей сети.

Для контроля состояния связей и соседних маршрутизаторов маршрутизаторы OSPF передают друг другу особые сообщения HELLO каждые 10 секунд. Небольшой объем этих сообщений делает возможным частое тестирование состояния соседей и связей с ними.

Каждый маршрутизатор посылает сообщения HELLO по групповому адресу 224.0.0.5. В этом сообщении он указывает свой идентификатор, а также информацию о своем непосредственном окружении, то есть IP-адрес интерфейса, через который сообщение HELLO передается в сеть, тип интерфейса — «точка-точка» или широковещательный, а также список соседей, о которых он узнал (на основании их сообщений HELLO).

Если сообщения HELLO перестают поступать от какого-либо непосредственного соседа, то маршрутизатор делает вывод о том, что состояние связи изменилось с работоспособного

на неработоспособное, после чего вносит соответствующие коррективы в свою топологическую базу данных. Одновременно он отсылает всем непосредственным соседям объявление LSA об этом изменении, которые также вносят исправления в свои базы данных и, в свою очередь, рассылают данное объявление LSA своим непосредственным соседям.

Результатом первого этапа является создания графа сети в виде **базы данных состояния связей LSDB** (Link State DataBase).

Этап 2 — нахождение оптимальных маршрутов и генерация таблицы маршрутизации. Задача нахождения оптимального пути на графе является достаточно сложной и трудоемкой. В протоколе OSPF для ее решения используется итеративный *алгоритм Дейкстры.*

Каждый маршрутизатор сети, действуя в соответствии с этим алгоритмом, ищет оптимальные маршруты от своих интерфейсов до всех известных ему подсетей. В каждом найденном таким образом маршруте запоминается только один шаг — до следующего маршрутизатора. Данные об этом шаге и попадают в таблицу маршрутизации. Таким образом, записи таблицы маршрутизации, созданные протоколом OSPF, имеют ту же структуру, что и записи, созданные протоколом RIP, что позволяет применять в одной и той же сети несколько различных протоколов маршрутизации. О том, как они взаимодействуют, рассказывается позже, в разделе «Маршрутизация в неоднородных сетях. Протокол BGP» этой главы.

Если состояние связей в сети изменилось и произошла корректировка графа сети, то каждый маршрутизатор заново ищет оптимальные маршруты и корректирует свою таблицу маршрутизации. Аналогичный процесс происходит и в том случае, когда в сети появляется новая связь или новый сосед, объявляющий о себе с помощью своих сообщений HELLO. При работе протокола OSPF конвергенция таблиц маршрутизации к новому согласованному состоянию происходит достаточно быстро — быстрее, чем в сетях, в которых работают дистанционно-векторные протоколы. Это время состоит из времени распространения по сети объявления LSA и времени работы алгоритма Дейкстры, обладающего быстрой сходимостью. Однако вычислительная сложность этого алгоритма предъявляет высокие требования к мощности процессоров маршрутизаторов. Если состояние сети не меняется, то объявления о связях не генерируются, топологические базы данных и таблицы маршрутизации не корректируются, что экономит пропускную способность сети и вычислительные ресурсы маршрутизаторов. Однако у этого правила есть исключение: каждые 30 минут маршрутизаторы OSPF обмениваются всеми записями базы данных топологической информации, то есть синхронизируют их для более надежной работы сети. Так как этот период достаточно большой, то данное исключение незначительно сказывается на загрузке сети.

Объявления о состоянии связей

В объявлении о связи LSA всегда сообщается информация о двух точках сети, которые эта связь соединяет.

Существуют несколько типов объявлений LSA.

Первый тип объявлений — *LSA Type 1* — распространяет информацию о маршрутизаторах и их связях, как между собой, так и о связях с IP-подсетями. Каждый маршрутизатор объявляет о себе, помещая в LSA Type 1 свой идентификатор (обычно свой loopback-адрес). Рассмотрим состав такого объявления на примере фрагмента сети, показанного на рис. 16.3. Будем считать, что маршрутизаторы идентифицируются такими адресами: R1 — 1.1.1.1, R2 — 2.2.2.2 и т. д.

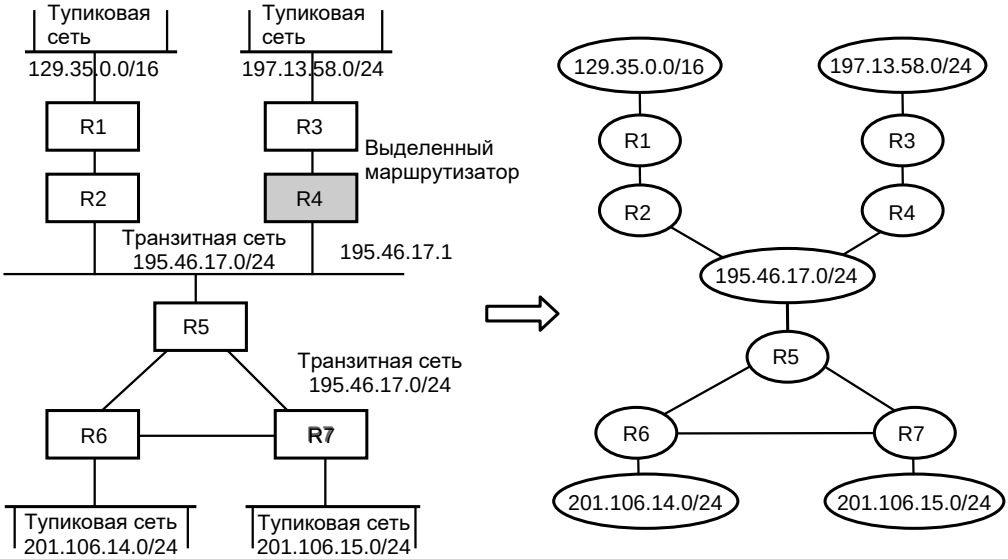


Рис. 16.3. Сеть и ее граф

Маршрутизатор R1 помещает в свое объявление LSA Type 1 следующую информацию:

- 1.1.1.1 — идентификатор маршрутизатора, который делает объявление;
- связь 1: связь типа «точка-точка» с маршрутизатором 2.2.2.2;
- связь 2: связь с тупиковой сетью 129.35.0.0/16.

В протоколе OSPF сети делятся на тупиковые и транзитные. К тупиковой сети присоединен только один маршрутизатор (например, к тупиковой сети 129.35.0.0 присоединен только маршрутизатор R1), а к транзитной — несколько (маршрутизаторы R2, R4 и R5 присоединены к транзитной сети 195.46.17.0/24). Для того чтобы топология сети, в которой имеются только тупиковые сети и связи «точка-точка» между маршрутизаторами, была правильно отражена в базе LSDB, маршрутизаторам достаточно обмениваться только объявлениями LSA Type 1. Но для того чтобы правильно построить топологию сети, включающую транзитные сети, таких объявлений недостаточно — они дополняются объявлениями LSA Type 2, которые передают детальную информацию о транзитной сети.

Объявления *LSA Type 2* может делать только один из маршрутизаторов, присоединенных к транзитной сети. Такой маршрутизатор называется ***назначенным*** (Dedicated Router, DR) для данной транзитной сети. Он представляет собой фиктивный узел сети, представляющий транзитную сеть. Маршрутизатор назначается транзитной сети не администратором. Он становится назначенным в результате выборов, в которых участвуют все маршрутизаторы, присоединенные к этой сети. О том, что нужно провести выборы, маршрутизаторы узнают из сообщений HELLO, получаемых друг от друга по транзитной сети: если у маршрутизатора оказывается больше одного соседа в этой сети (и она помечена как широковещательная), то выборы необходимы. Выигрывает маршрутизатор с бóльшим значением приоритета (задается администратором) или в случае равенства приоритетов — с бóльшим значением идентификатора.

Назначенный маршрутизатор начинает распространять объявление LSA Type 2 о транзитной сети, указывая свой идентификатор и адрес своего интерфейса на ней в качестве идентификатора сети, а также ее маску. Все маршрутизаторы, присоединенные к данной транзитной сети, объявляют о своей связи с ней в форме пары «идентификатор маршрутизатора — адрес интерфейса DR в транзитной сети», так как назначенный маршрутизатор представляет собой транзитную сеть. На основании объявлений двух типов — LSA Type 2 от маршрутизатора DR и LSA Type 1 от остальных маршрутизаторов — и строится граф данного сегмента общей сети. В нашем примере маршрутизатор R4 является назначенным маршрутизатором для транзитной сети 195.46.17.0/24, о которой он объявляет сообщением LSA Type 2 с указанием ее префикса и маски. Он также перечисляет в своем объявлении маршрутизаторы R2 и R5, присоединенные к этой сети. Маршрутизаторы объявляют о связи с интерфейсом 195.46.17.1 назначенного маршрутизатора R4 с помощью объявлений LSA Type 1, тем самым объявляя о связи с сетью 195.46.17.0.24, которую представляет назначенный маршрутизатор.

Метрики

При поиске оптимальных маршрутов протокол OSPF по умолчанию использует метрику, учитывающую пропускную способность каналов связи. Кроме того, допускается применение двух других метрик, учитывающих задержки и надежность передачи пакетов каналами связи. Для каждой из метрик протокол OSPF строит *отдельную* таблицу маршрутизации. Выбор нужной таблицы происходит в зависимости от значений битов TOS в заголовке пришедшего IP-пакета. Если в пакете бит D (Delay — задержка) установлен в 1, то для этого пакета маршрут должен выбираться из таблицы, в которой содержатся маршруты, имеющие минимальную задержку. Аналогично пакет с установленным битом T (Throughput — пропускная способность) должен маршрутизироваться по таблице, построенной с учетом пропускной способности каналов, а установленный в единицу бит R (Reliability — надежность) указывает на то, что должна использоваться таблица, для построения которой критерием оптимизации служит надежность доставки.

Протокол OSPF поддерживает стандартные для многих протоколов (например, для протокола покрывающего дерева) значения расстояний для метрики, отражающей пропускную способность: так, для сети Ethernet она равна 10, для Fast Ethernet — 1, для канала, обладающего пропускной способностью 1,544 Мбит/с, — 65, для канала с пропускной способностью 56 Кбит/с — 1785. При наличии высокоскоростных каналов (Gigabit Ethernet, 10G Ethernet) администратору нужно задать другую шкалу скоростей, назначив единичное расстояние наиболее скоростному каналу.

Протокол OSPF разрешает хранить в таблице маршрутизации несколько маршрутов к одной сети, если они обладают равными метриками. В подобных случаях маршрутизатор может работать в режиме баланса загрузки маршрутов, отправляя пакеты попеременно по каждому из маршрутов.

Области сети

Вычислительная сложность протокола OSPF быстро растет с увеличением размера сети. Для преодоления этого недостатка в протоколе OSPF вводится понятие **области сети**.

Область сети OSPF — это непрерывное подмножество маршрутизаторов и их интерфейсов, в котором все маршрутизаторы используют общую базу данных LSDB для нахождения маршрутов. Область должна быть физически непрерывной, чтобы внутренняя топология была полностью видна всем маршрутизаторам внутри нее.

Маршрутизаторы, принадлежащие некоторой области, строят граф связей только для этой области, что упрощает задачу. Между областями информация о связях не передается, а пограничные для областей маршрутизаторы, называемые маршрутизаторами **ABR** (Area Border Router), обмениваются только информацией об адресах сетей, имеющихся в каждой из областей, и *расстоянием от пограничного маршрутизатора до каждой сети*. То есть фактически маршрутизация между областями OSPF осуществляется на основе дистанционно-векторного алгоритма. Для передачи объявлений между областями используются специальные, так называемые *суммарные* маршрутные объявления *LSA Type 3*, так как они передают только суммарную информацию об области, а не ее точную топологию. При передаче пакетов между областями выбирается один из пограничных маршрутизаторов области, а именно тот, у которого расстояние до нужной сети меньше.

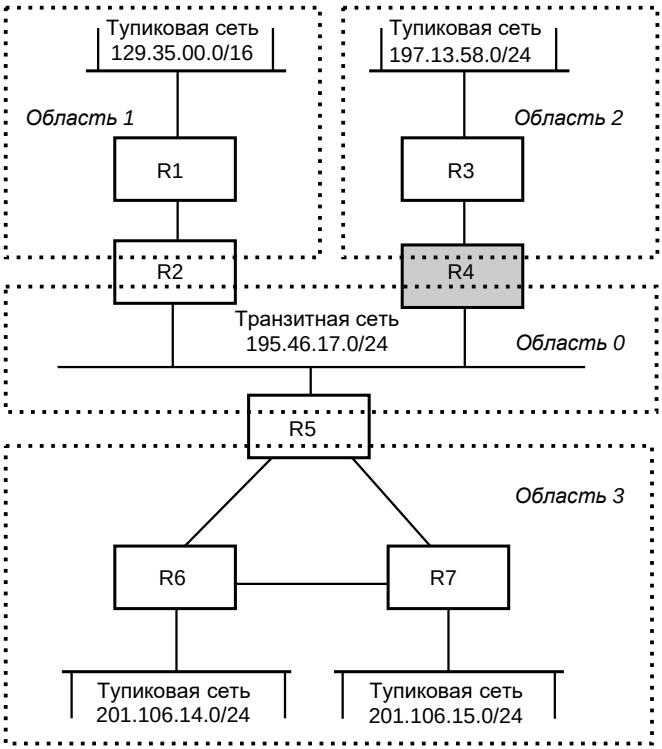


Рис. 16.4. Разбиение сети OSPF на области

На рис. 16.4 показано разделение сети рис. 16.3 на четыре области. Маршрутизаторы R2, R4 и R5 являются пограничными маршрутизаторами ABR, поскольку они имеют интерфейсы в двух областях каждый. При разбиении сети на области одна из областей, а именно область 0, должна быть магистральной. Все остальные области могут быть связаны только с ней, но

не между собой. Такая древовидная топология областей OSPF решает задачу зацикливания (петель) межобластных маршрутных объявлений LSA Type 3.

В заключение дадим краткую характеристику основному сопернику протокола OSPF — протоколу IS-IS. Протокол **IS-IS** (Intermediate System to Intermediate System) функционально близок к протоколу OSPF. Существуют различные версии IS-IS, рассчитанные на работу в различных стеках протоколов и способные переносить в своих сообщениях адресную информацию различного типа, например версия IS-IS для стека TCP/IP или версия IS-IS для работы в сетях Ethernet.

Маршрутизация в неоднородных сетях. Протокол BGP

Взаимодействие протоколов маршрутизации

В одной и той же сети могут одновременно работать несколько разных протоколов маршрутизации (рис. 16.5). Это означает, что на некоторых (не обязательно на всех) маршрутизаторах сети установлено и функционирует несколько протоколов маршрутизации, но при этом, естественно, через сеть взаимодействуют только одноименные протоколы. То есть если маршрутизатор 1 поддерживает, например, протоколы RIP и OSPF, маршрутизатор 2 — только RIP, а маршрутизатор 3 — только OSPF, то маршрутизатор 1 будет взаимодействовать с маршрутизатором 2 по протоколу RIP, с маршрутизатором 2 — по OSPF, а маршрутизаторы 2 и 3 вообще непосредственно друг с другом взаимодействовать не смогут.

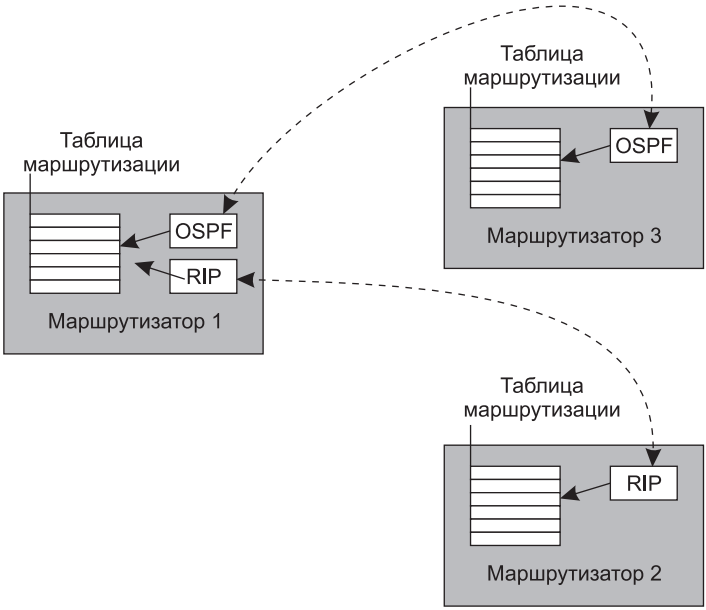


Рис. 16.5. Применение нескольких протоколов маршрутизации в одной сети

В маршрутизаторе, поддерживающем одновременно несколько протоколов, каждая запись в таблице является результатом работы одного из этих протоколов. Если информация о некоторой сети появляется от нескольких протоколов, то для однозначности выбора маршрута (а данные разных протоколов могут вести к разным рациональным маршрутам) устанавливаются *приоритеты протоколов маршрутизации*. Обычно предпочтение отдается протоколам LSA как располагающим более полной информацией о сети по сравнению с протоколами DVA. В некоторых ОС в формах вывода на экран и печать в каждой записи таблицы маршрутизации имеется отметка о том, с помощью какого протокола маршрутизации эта запись получена. Но даже если эта отметка на экран и не выводится, она обязательно имеется во внутреннем представлении таблицы маршрутизации. По умолчанию каждый протокол маршрутизации, работающий на определенном маршрутизаторе, распространяет только собственную информацию, то есть ту, что была получена этим маршрутизатором по данному протоколу. Так, если о маршруте к некоторой сети маршрутизатор узнал по протоколу RIP, то и распространять по сети объявления об этом маршруте он будет с помощью протокола RIP.

Однако такой «избирательный» режим работы маршрутизаторов ставит невидимые барьеры на пути распространения маршрутной информации, создавая в составной сети области взаимной недостижимости. Задача маршрутизации решается эффективнее, если маршрутизаторы смогут обмениваться маршрутной информацией, полученной разными протоколами маршрутизации. Такая возможность реализуется в особом режиме работы маршрутизатора, называемом **режимом перераспределения маршрутов**. Этот режим позволяет одному протоколу маршрутизации использовать не только свои, но и чужие записи таблицы маршрутизации, полученные с помощью другого протокола маршрутизации, указанного при конфигурировании.

Как видим, применение нескольких протоколов маршрутизации даже в пределах небольшой составной сети — дело непростое, поскольку администратору требуется провести определенную работу по конфигурированию каждого маршрутизатора. Очевидно, что для крупных составных сетей нужно качественно иное решение.

Внутренние и внешние шлюзовые протоколы

Такое решение было найдено для самой крупной на сегодня составной сети — интернета. Это решение базируется на понятии автономной системы.

Автономная система (Autonomous System, **AS**) — это совокупность сетей под единым административным управлением, обеспечивающим общую для всех входящих в автономную систему маршрутизаторов политику маршрутизации.

Обычно автономной системой управляет один поставщик услуг интернета, самостоятельно выбирая, какие протоколы маршрутизации должны использоваться в некоторой автономной системе и каким образом между ними должно выполняться перераспределение маршрутной информации. Крупные поставщики услуг и корпорации могут представить свою составную сеть как набор нескольких автономных систем.

Регистрация автономных систем происходит централизованно, как и регистрация IP-адресов и DNS-имен. Номер автономной системы состоит из 32 двоичных разрядов и никак не связан с префиксами IP-адресов входящих в нее сетей. В соответствии с этой концепцией интернет выглядит как набор взаимосвязанных автономных систем, каж-

дая из которых состоит из взаимосвязанных сетей (рис. 16.6), соединенных **внешними шлюзами**.

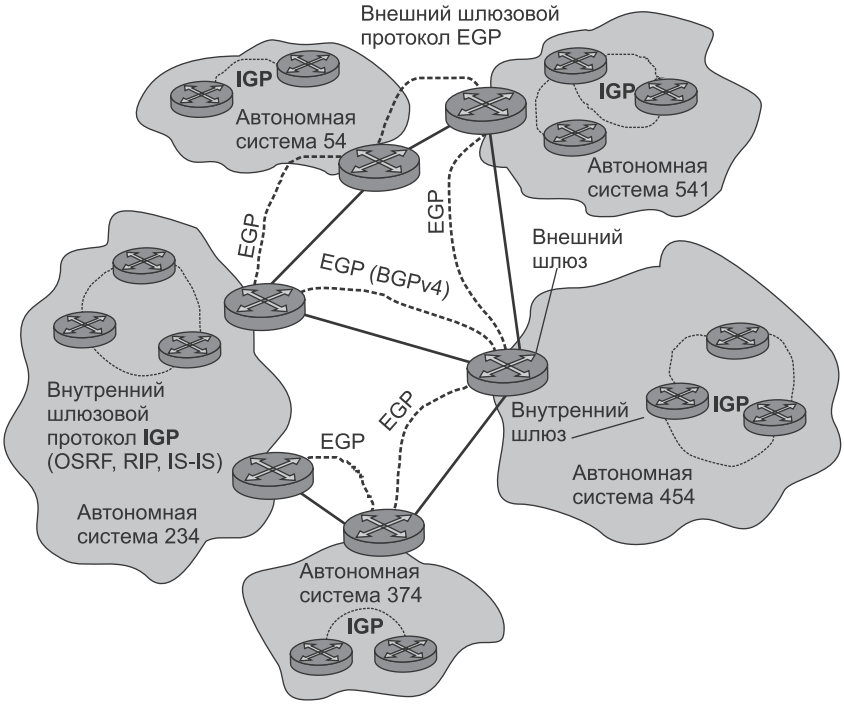


Рис. 16.6. Автономные системы Интернета

Основная цель деления интернета на автономные системы — обеспечение многоуровневого подхода к маршрутизации. До введения автономных систем предполагался двухуровневый подход, то есть сначала маршрут определялся как *последовательность сетей*, а затем вел непосредственно к заданному узлу в конечной сети (именно этот подход мы использовали до сих пор). С появлением автономных систем появляется третий, верхний, уровень маршрутизации: сначала маршрут определяется как *последовательность автономных систем*, затем — как последовательность сетей и только потом ведет к конечному узлу. Такой подход делает процедуру маршрутизации масштабируемой.

Выбор маршрута между автономными системами осуществляют внешние шлюзы, использующие особый тип протокола маршрутизации — **внешний шлюзовой протокол** (Exterior Gateway Protocol, EGP). Сейчас для работы в такой роли сообщество интернета утвердило стандартный **пограничный шлюзовой протокол** версии 4 (Border Gateway Protocol, **BGPv4**). В качестве адреса следующего маршрутизатора в протоколе BGPv4 указывается адрес точки входа в соседнюю автономную систему.

За *маршрут внутри автономной системы* отвечают **внутренние шлюзовые протоколы** (Interior Gateway Protocol, **IGP**). К числу IGP относятся знакомые нам протоколы RIP, OSPF и IS-IS. В случае транзитной автономной системы эти протоколы указывают точную последовательность маршрутизаторов от точки входа в автономную систему до точки выхода из нее.

ПРИМЕЧАНИЕ

Внутри каждой автономной системы может применяться любой из существующих протоколов маршрутизации, в то время как между автономными системами всегда применяется один и тот же протокол, являющийся своеобразным языком «эсперанто», на котором автономные системы общаются между собой.

Концепция автономных систем скрывает от администраторов магистралей интернета проблемы маршрутизации пакетов на более низком уровне — уровне сетей. Для администратора магистрали не важно, какие протоколы маршрутизации применяются внутри автономных систем, поскольку для него существует единственный протокол маршрутизации — BGP.

Протокол BGP

Пограничный (внешний) шлюзовой протокол (Border Gateway Protocol, **BGP**) в версии **4** является сегодня основным протоколом обмена маршрутной информацией между *автономными системами* интернета. Характерными его особенностями является то, что он работает на основе:

- политики (policy based);
- вектора пути (path vector).

BGP успешно работает при любой топологии связей между автономными системами, что соответствует современному состоянию интернета.

Поясним основные принципы работы BGP на примере (рис. 16.7). В каждой из трех автономных систем (AS 1021, AS 363 и AS 520) имеется несколько маршрутизаторов, исполняющих роль внешних шлюзов. На каждом из них работает протокол BGP, с помощью которого они общаются между собой.

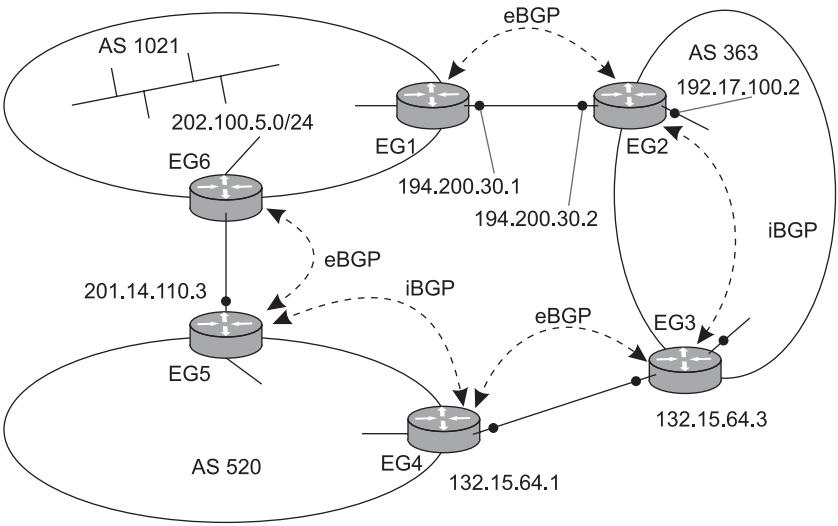


Рис. 16.7. Поиск маршрута между автономными системами с помощью протокола BGP

Маршрутизатор взаимодействует с другими маршрутизаторами по протоколу BGP только в том случае, если администратор *явно* указывает при конфигурировании, что эти маршрутизаторы являются его *соседями*. Например, маршрутизатор EG1 в рассматриваемом примере будет взаимодействовать по протоколу BGP с маршрутизатором EG2 не потому, что эти маршрутизаторы соединены двухточечным каналом, а потому, что при конфигурировании маршрутизатора EG1 в качестве соседа ему был указан маршрутизатор EG2 (с адресом 194.200.30.2). Аналогично, при конфигурировании маршрутизатора EG2 его соседом был назначен маршрутизатор EG1 (с адресом 194.200.30.1).

Такой способ взаимодействия удобен в ситуации, когда маршрутизаторы, обменивающиеся маршрутной информацией, принадлежат разным поставщикам услуг (Internet Service Provider, ISP). Администратор ISP может решать, с какими автономными системами он будет обмениваться трафиком, а с какими нет, задавая список соседей для своих внешних шлюзов.

Протоколы RIP и OSPF, разработанные для применения внутри автономной системы, обмениваются маршрутной информацией со всеми маршрутизаторами, находящимися в пределах их непосредственной досягаемости (по локальной сети или через двухточечный канал). Это означает, что информация обо всех сетях появляется в таблице маршрутизации каждого маршрутизатора, так что каждая сеть оказывается достижимой для каждой. В корпоративной сети это нормальная ситуация, а в сети провайдера нужно поддерживать разнообразные и гибкие **политики маршрутизации**, говорящие о том, каким соседям передавать маршрутные объявления и о каких сетях им в этих объявлениях сообщать, а также от каких соседей и о каких сетях можно принимать маршрутные объявления. Политика маршрутизации провайдера отражает условия по взаимной передаче трафика, имеющиеся в *пиринговых соглашениях* (от *peering* — отношения равных субъектов), которые провайдер заключает с другими провайдерами.

Для установления сеанса с указанными соседями маршрутизаторы BGP используют протокол TCP (порт 179). При установлении BGP-сеанса могут применяться разнообразные способы аутентификации маршрутизаторов, повышающие безопасность работы автономных систем. Основное сообщение протокола BGP — сообщение UPDATE (обновить), с помощью которого маршрутизатор сообщает маршрутизатору соседней автономной системы о достижимости сетей, относящихся к его собственной автономной системе. Само название этого сообщения говорит о том, что это триггерное объявление, посылаемое соседу, если в топологии автономной системы происходят изменения: появляются новые сети или новые пути к сетям либо же, напротив, исчезают существовавшие сети или пути. В одном сообщении UPDATE можно объявить об одном новом маршруте или аннулировать несколько маршрутов, переставших существовать.

BGP является протоколом маршрутизации на основе **вектора пути** (path vector), в котором под маршрутом понимается *последовательность автономных систем*, которую нужно пройти на пути к указанной в адресе сети. Каждое объявление маршрута содержит путь. Например, сообщение BGP о маршруте может выглядеть следующим образом: «Чтобы достичь сети 201.36.14.0 надо пройти путь: AS 100 → AS 200 → AS 300».

Это отличает протокол BGP от **дистанционно-векторного** (distance vector) протокола RIP, который передает соседям сообщение, содержащее сеть назначения и метрику, например: «Сеть с адресом 201.36.14.0 находится от меня на расстоянии трех хопов».

Формальная запись о маршруте (BGP Route) к сети (Network/Mask_length) выглядит так:

BGP Route = AS_Path; NextHop; Network/Mask_length;

Здесь AS_Path — набор номеров автономных систем, NextHop — IP-адрес маршрутизатора, через который нужно передавать пакеты в сеть Network/Mask_length. Например, если маршрутизатор EG1 хочет объявить маршрутизатору EG2 о том, что в AS 1021 появилась новая сеть 202.100.5.0/24, для достижения которрой следующим хопом является узел с адресом 194.200.30.1, то он формирует такое сообщение:

BGP Route = **AS 1021; 194.200.30.1; 202.100.5.0/24**.

Затем он передает это сообщение маршрутизатору EG2 автономной системы AS 363 (с которым у него предварительно должен быть установлен BGP-сеанс).

Список сетей своей AS, которые протокол BGP предает своим соседям в объявлениях UPDATE, может формироваться двумя способами. Динамический способ основан на перераспределении маршрутов внутреннего протокола маршрутизации, например, RIP, в объявления BGP. В том случае, если администратор AS хочет передавать другим AS информацию только о некоторых своих сетях, то он может их указать непосредственно в конфигурации протокола BGP, а режим перераспределения маршрутов не использовать.

Маршрутизатор EG2, получив сообщение UPDATE, запоминает в своей таблице маршрутизации информацию о сети 202.100.5.0/24 вместе с адресом следующего маршрутизатора 194.200.30.1 и отметкой о том, что эта информация была получена по протоколу BGP. Маршрутизатор EG2 обменивается маршрутной информацией с внутренними шлюзами системы AS 363 по какому-либо протоколу группы IGP, например OSPF. Если у EG2 установлен *режим перераспределения маршрутов* BGP в маршруты OSPF, то все внутренние шлюзы AS 363 узнают о существовании сети 202.100.5.0/24 с помощью объявления OSPF, которое будет внешним. В качестве адреса следующего маршрутизатора маршрутизатор EG2 начнет теперь объявлять адрес собственного внутреннего интерфейса, например 192.17.100.2.

Обычно протокол BGP не перераспределяет все внешние маршруты во внутренний протокол IGP, потому что это может привести к ненужному переполнению таблиц IGP-маршрутизаторов всем множеством префиксов сетей, существующих в интернете. Вместо использования режима перераспределения маршрутов к внешним сетям можно применить другой способ указания маршрута к внешним сетям — задать внутренним маршрутизаторам маршрут по умолчанию к одному или нескольким маршрутизаторам BGP, аналогично тому, как указывается маршрутизатор по умолчанию для хоста сети. Другой способ ограничения переполнения таблиц маршрутизации внутренних IGP-маршрутизаторов состоит в том, что администратор автономной системы явно указывает маршрутизатору BGP, какие префиксы внешних сетей он должен перераспределять в определенный внутренний протокол маршрутизации. Такое действие осуществляется с помощью одного из правил политики, управляющих поведением BGP-маршрутизатора.

Но для распространения сообщения о сети 202.100.5.0/24 в другие автономные системы, например в AS 520, протокол OSPF использоваться не может. Маршрутизатор EG3, связанный с маршрутизатором EG4 автономной системы 520, должен применять протокол BGP, генерируя сообщение UPDATE нужного формата. Для решения этой задачи он не может задействовать информацию о сети 202.100.5.0/24, полученную от протокола OSPF через один из своих внутренних интерфейсов, так как она имеет другой формат и не содержит, например, сведений о номере автономной системы, в которой находится эта сеть.

Проблема решается за счет того, что маршрутизаторы EG2 и EG3 также устанавливают между собой BGP-сеанс, хотя они и принадлежат одной и той же автономной системе. Такая реализация протокола BGP называется **внутренней версией BGP** (Interior BGP, **iBGP**), в отличие от основной, **внешней версии** (Exterior BGP, **eBGP**). В результате маршрутизатор EG3 получает нужную информацию от маршрутизатора EG2 и передает ее внешнему соседу — маршрутизатору EG4. При формировании нового сообщения UPDATE маршрутизатор EG3 трансформирует сообщение, полученное от маршрутизатора EG2, добавляя в список автономных систем собственную автономную систему AS 520, а полученный адрес следующего маршрутизатора заменяет адресом собственного интерфейса:

AS 363, AS 1021; 132.15.64.3; 202.100.5.0/24.

Номера автономных систем позволяют исключить зацикливание сообщений UPDATE. Например, когда маршрутизатор EG5 передаст маршрутизатору EG6 следующее сообщение о сети 202.100.5.0/24:

AS 520, AS 363, AS 1021; 201.14.110.3; 202.100.5.0/24,

последний не станет его использовать, так как в списке автономных систем уже имеется номер его собственной автономной системы, из чего следует, что сообщение зациклилось.

Подход на основе вектора пути устраняет некоторые ключевые проблемы дистанционно-векторной маршрутизации в масштабах интернета.

Во-первых, как было показано выше, автоматически решается проблема петель и зацикливания пакетов. Действительно, маршрутизатор имеет перед собой весь путь, поэтому если он обнаруживает в пути номер своей собственной AS, то маршрут отклоняется. Это делает маршрутизацию на основе вектора пути изначально беспетлевой, без необходимости использования метода расщепления горизонта, замораживания изменений и других рассмотренных ранее методов борьбы с ложными маршрутами.

Во-вторых, вектор пути строится на основе политики, а не на основе метрик (количество хопов, задержка и т. д.), что делает протокол BGP более эффективным в вычислительном плане. Путь BGP не содержит поля метрики, хотя в качестве критерия выбора маршрута можно было бы использовать количество промежуточных автономных систем. Однако обычно качество пути оценивается не значением его длины (стоимости), а тем, насколько он соответствует выбранной политике. Это значит, что более длинный (дорогой) маршрут, удовлетворяющий критериям политики, может оказаться предпочтительнее более короткого, но не соответствующего этим критериям маршрута. Выбрать маршрут на основе нескольких правил политики гораздо проще, чем найти кратчайший маршрут на графе с помощью алгоритма Дейкстры, поэтому протокол BGP не требует высоких вычислительных затрат.

В-третьих, использование вектора пути делает маршрутизацию BGP масштабируемой в пределах глобального интернета в то время, как дистанционно-векторный протокол RIP и протокол состояния OSPF поддерживают не более 100 000 префиксов и 60 000 автономных систем.

Номера автономных систем в сообщении UPDATE позволяют избегать маршрутные петли при передаче сообщений между автономными системами по протоколу EBGP. Однако при передаче сообщений UPDATE между несколькими маршрутизаторами одной и той же AS

(то есть маршрутизаторами с IBGP-связями) этот механизм не работает, так как при такой передаче номера AS в сообщении UPDATE не меняются — передачи идут в пределах одной и той же AS. Чтобы избежать зацикливания, маршрутизаторы BGP придерживаются следующего правила: сообщение, полученное по соединению IBGP, никогда не передается другим IBGP этой AS, оно передается только EBGP-маршрутизаторам других AS.

Следование этому правилу вынуждает создавать полносвязную топологию между всеми IBGP автономной системы. Если такой топологии нет, как это показано на рис. 16.8, *а*, то сообщение UPDATE не достигает всех маршрутизаторов IBGP, а значит, не передается в некоторые соседние AS. На рисунке *а* в таком положении оказался маршрутизатор EG21, который не получил сообщения о сети 202.100.5.0/24 от маршрутизатора EG3, потому что у него нет с ним IBGP-соединения.

На рисунке *б* проблема решена: полносвязная топология создана и сообщение от маршрутизатора EG3 теперь достигает трех остальных IBGP маршрутизатора AS 363. То же самое будет происходить и с сообщениями, полученными маршрутизаторами EG2, EG21 и EG31: они будут распространены между всеми остальными IBGP-маршрутизаторами AS 363.

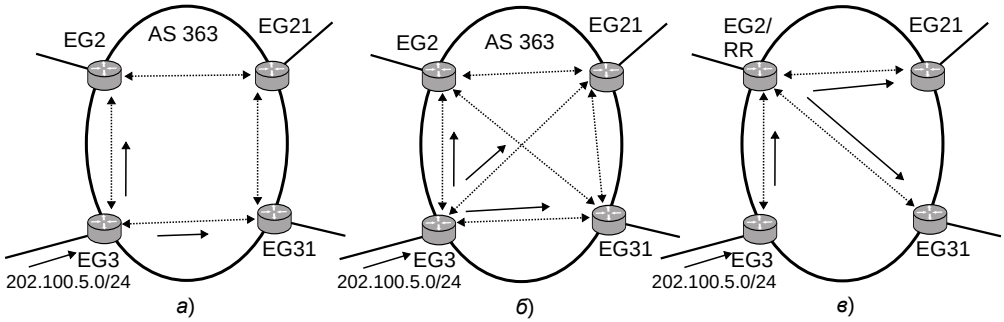


Рис. 16.8. Передача сообщения UPDATE между IBGP-маршрутизаторами

Найденное решение имеет один недостаток: оно плохо масштабируется. Если в большой автономной системе имеются десятки маршрутизаторов BGP, то полносвязная топология требует слишком большого количества IBGP-соединений. Более масштабируемым решением является применение в AS выделенного IBGP-маршрутизатора, называемого **рефлектором маршрутов** (Route Reflector, RR). На рис. 16.8, *в* таким маршрутизатором является маршрутизатор EG2. Все остальные BGP-маршрутизаторы сети устанавливают IBGP-соединения с рефлектором маршрутов в качестве его клиентов. Рефлектору маршрутов разрешается нарушать правило «не передавать сообщение, полученное от IBGP соседа другим IBGP-маршрутизаторам сети». Поэтому маршрутизатор EG2/RR, получив сообщение о сети 202.100.5.0/24 от маршрутизатора EG3, передает его своим клиентам EG21 и EG31. Клиенты общаются по соединениям iBGP только с рефлектором маршрутов, между собой у них соединений iBGP нет.

Применение рефлекторов маршрутов является обычной практикой провайдеров интернета. Для обеспечения надежности автономная система обычно имеет несколько рефлекторов маршрутов. Рассмотрение организации работы такой сети выходит за рамки темы данной книги.

В главе 31 можно прочитать об уязвимостях и защите протокола BGP.

Групповое вещание

Групповое вещание (multicast), применяемое ранее только в радио- и телевизионных сетях, в последние годы все шире внедряется в компьютерные сети. Наиболее востребована эта технология в интернете, который представляет собой идеальную среду для массового распространения по подписке мультимедийной информации: аудиозаписей, видеофильмов, информационных дайджестов и т. п.

Стандартная модель группового вещания IP

Основной целью группового вещания является создание эффективного механизма передачи данных от одного источника нескольким получателям. Для решения этой задачи могут использоваться несколько подходов, например, индивидуальная рассылка, широковещательная рассылка, привлечение сервисов прикладного уровня.

При *индивидуальной рассылке* (unicast) на основе уникальных адресов источник данных, которые надо доставить некоторой группе узлов, генерирует их в количестве экземпляров, равном количеству узлов-получателей, состоящих в данной группе. То есть передача по принципу «один ко многим» сводится к нескольким передачам «один к одному». Очевидно, что передача нескольких идентичных копий на участках, где маршруты к разным членам группы перекрываются (это особенно характерно для начальных участков), приводит к избыточному трафику.

При *широковещательной рассылке* (broadcast) для того, чтобы доставить данные группе узлов-получателей, источник генерирует один экземпляр данных, но снабжает этот экземпляр широковещательным адресом, который диктует маршрутизаторам сети копировать данные и рассылать их всем конечным узлам независимо от того, заинтересованы узлы в получении этих данных или нет. В этом случае, как и в предыдущем, существенная доля трафика является избыточной.

В случае *привлечения сервисов прикладного уровня* источник генерирует один экземпляр данных и, используя индивидуальный адрес, передает данные одному из членов группы, который генерирует копию и направляет ее другому члену группы и т. д. Перевод решения задачи с нижних транспортных уровней на прикладной уровень повышает суммарные накладные расходы сети на реализацию групповой доставки и делает этот механизм менее гибким.

Таким образом, традиционные механизмы доставки пакетов стека TCP/IP малопригодны для поддержки группового вещания. В такой ситуации наиболее эффективным решением является использование специально разработанного механизма группового вещания, ориентированного на сокращение избыточного трафика и накладных расходов сети.

Главная идея группового вещания состоит в следующем: источник генерирует только один экземпляр сообщения с *групповым адресом*, которое по мере перемещения по сети копируется на каждой из «развилок», ведущих к тому или иному члену группы, указанной в адресе данного сообщения. В конце концов, пакет с групповым адресом достигает маршрутизатора, к которому *непосредственно* подключена сеть с хостами — членами данной группы. Напомним, у хостов, относящихся к той или иной группе, интерфейс наряду с индивидуальным адресом имеет еще один или несколько групповых адресов — адресов класса D — по числу групп, в которых состоит данный хост. Как и в случае обычной

маршрутизации на базе индивидуальных адресов, маршрутизатор упаковывает пакет с групповым адресом в кадр канального уровня (той технологии, которая используется в данной локальной сети, например Ethernet), снабжая его *групповым MAC-адресом*, соответствующим групповому IP-адресу пакета[1]. Кадр с пакетом группового вещания поступает в локальную сеть, распознается и захватывается интерфейсами хостов, являющихся членами данной группы. При таком подходе данные рассылаются только тем узлам, которые заинтересованы в их получении. Функция репликации группового сообщения и продвижения копий в сторону членов группы возлагается на маршрутизаторы, для чего *они должны быть оснащены соответствующими программно-аппаратными средствами*. Такой режим экономит пропускную способность за счет передачи только того трафика, который необходим.

Стив Диринг (Steve Deering) — один из главных идеологов группового вещания — сформулировал несколько принципиальных положений, регламентирующих поведение конечных узлов сети, которые являются источниками и получателями группового трафика.

- *Дейтаграммный подход.* Источник может посылать пакеты UDP/IP в любое время без необходимости регистрировать или планировать передачи, реализуя сервис «по возможности».
- *Открытые группы.* Источники должны знать только групповой адрес. Они не должны знать членов группы и не обязательно должны быть членами той группы, которой посылают данные. Группа может быть образована узлами, принадлежащими к разным IP-сетям и подсетям, иметь любое число источников данных.
- *Динамические группы.* Хосты могут присоединяться к группам или покидать группы без необходимости регистрации, синхронизации или переговоров с каким-либо централизованным элементом группового управления. Членство в группе является динамическим — хосты могут присоединиться к группе или выйти из группы в любой момент времени, к тому же они могут быть членами нескольких групп.

Адреса группового вещания

Ранее в главе 13, изучая типы IP-адресов, мы отмечали, что адреса IPv4 из диапазона 224.0.0.0–239.255.255.255 относятся к классу D и зарезервированы для группового вещания. Эти адреса используются для идентификации групп, источников, а также для административных нужд группового вещания. В общем случае адреса используются динамически — если после остановки вещания источник снова начинает передачу, то он может задействовать новый адрес группового вещания. Так называемые *хорошо известные* источники обычно наделяются постоянным групповым адресом. Информацию о том, какие адреса уже закреплены для исполнения некоторой постоянной роли, а также о том, как использовать адресное пространство адресов класса D, дает документ RFC 5771 полномочной организации по цифровым адресам интернета IANA.

(S) *Структурирование адресного пространства группового вещания*

---

[1] Об отображении групповых IP-адресов на групповые MAC-адреса см. далее в разделе «Протокол IGMP».

Протокол IGMP

На основе описанной концепции для стека TCP/IP был разработан ряд протоколов, с помощью которых можно организовать групповое вещание. Эти протоколы делятся на две категории. В первую входит **протокол управления группами в интернете** (Internet Group Management Protocol, **IGMP**), с помощью которого, во-первых, хосты сообщают маршрутизатору о своем «желании» присоединиться к некоторой группе, во-вторых, маршрутизатор собирает информацию о принадлежности хостов в непосредственно подключенных к нему подсетях к той или иной группе.

Ко второй категории относятся **протоколы маршрутизации группового вещания** (multicast routing protocols), которые необходимы для продвижения через сеть произвольной конфигурации пакетов, несущих в себе информацию для групповых получателей. Маршрутизаторы, которые поддерживают эти протоколы маршрутизации, называются **маршрутизаторами группового вещания** (multicast router). Протоколы маршрутизации группового вещания **DVMRP**, **MOSPF** и **PIM** опираются на разные подходы, но в конечном итоге все они сводятся к построению покрывающего дерева, связывающего все хосты в определенной группе. Протоколы маршрутизации группового вещания осуществляют постоянный мониторинг покрывающего дерева и время от времени отсекают ветви дерева, которые из-за изменения состояния сети уже не ведут к членам той или иной группы.

Подобно протоколу ICMP, протокол IGMP упаковывает свои сообщения в пакеты IP. Информация о том, что хост хочет стать членом некоторой группы, поступает на уровень IP от приложений. Например, если на хосте работает приложение интернет-телевидения, то оно может быть сконфигурировано так, чтобы получать информацию от определенных источников, которые вещают на определенные групповые адреса. Протокол IGMP является *асимметричным* протоколом, определяющим взаимодействие непосредственно связанных друг с другом хоста и маршрутизатора группового вещания. Одна его часть (*оповещение маршрутизатора*) регламентирует поведение хостов, которые с помощью IGMP оповещают маршрутизатор о своем участии в той или иной группе, а другая (*опрос членов группы*) — поведение маршрутизаторов, которые делают запросы к хостам и, основываясь на полученной информации, определяют, в какие подключенные к ним сети необходимо передавать групповой трафик. Начиная с первой версии, протокол IGMP использует два основных типа сообщений:

- запрос о членстве (membership query) — сообщение маршрутизатора;
- отчет о членстве (membership report) — сообщение хоста.

С развитием функциональности протокола IGMP набор сообщений расширялся: появился новый тип сообщения — *покинуть группу*, а также различные варианты запросов, о которых будет сказано ниже. Опрос членов группы выполняет только один из маршрутизаторов подсети, называемый *доминирующим* маршрутизатором. В IGMPv3 на эту роль выбирается маршрутизатор с наименьшим IP-адресом. Доминирующий маршрутизатор периодически опрашивает все хосты (групповой адрес 224.0.0.1)[1] в непосредственно присоединенных к нему подсетях, проверяя, активны ли члены всех известных ему групп. Остальные (не

[1] Сообщения с групповым адресом 224.0.0.1 обрабатываются специальным образом без участия IGMP и других протоколов группового вещания.

выбранные) маршрутизаторы прослушивают сеть и, если обнаруживают отсутствие сообщений-запросов в течение некоторого периода (обычно 250 секунд), то повторяют процедуру выбора нового доминирующего маршрутизатора.

С помощью сообщения *запрос о членстве* маршрутизатор пытается узнать, в каких группах состоят хосты в локальной сети, присоединенной к какому-либо его интерфейсу. Запрос о членстве существует в нескольких вариантах:

- **Общий запрос** (General Query) — периодически рассылается маршрутизатором всем хостам, подключенным к его сети. На этот запрос члены *всех групп* сообщают о своем членстве.
- **Запрос о конкретной группе** (Group-Specific Query) — посылается по соответствующему групповому адресу для того, чтобы определить состояние подписки именно для этой группы.
- **Запрос о конкретных группе и источнике** (Group-and-Source-Specific Query) — заключается в том, какие из указанных в запросе источников интересуют каждого из членов данной конкретной группы, которой адресован запрос.

Маршрутизатор регулярно посылает запросы о членстве (по умолчанию — каждые 125 секунд), чтобы проверить, имеются ли в каждой группе еще члены. На каждом из интерфейсов с установленными средствами IGMP маршрутизаторами поддерживаются кэш-таблицы групп. Кэш-таблица содержит список всех групп, в составе которых есть хотя бы один член. Для каждой строки таблицы установлен тайм-аут. Если для некоторой группы ответ не поступает в течение установленного для нее тайм-аута, то соответствующая строка удаляется из кэш-таблицы, и маршрутизатор считает, что членов этой группы в сети больше нет.

Сообщением *отчет о членстве* хосты отвечают маршрутизатору, который послал в сеть запрос о членстве. В сообщении содержится информация об адресе группы, в которой они состоят. Для маршрутизатора, получающего ответные сообщения, важен только факт наличия членов той или иной группы (групп), а не принадлежность конкретных хостов конкретным группам. Этот факт будет использован другими маршрутизаторами группового вещания сети для продвижения пакетов группового вещания в ту часть сети, за которую отвечает данный маршрутизатор. Хост посылает маршрутизатору IGMP-сообщение *отчет о членстве* не только в ответ на запрос маршрутизатора, но и *по собственной инициативе*, когда пытается присоединиться к определенной группе. После такого сообщения хост может рассчитывать на то, что трафик для этой группы действительно будет доставляться в сеть, к которой этот хост принадлежит.

Сообщение **покинуть группу** (leave group) используется хостом, чтобы сигнализировать о желании покинуть некоторую группу, в которой он до этого состоял. Получив это сообщение, маршрутизатор посылает специфический запрос о членстве членам только этой конкретной группы, и если не получает на него ни одного ответа (что говорит от том, что это последний хост в группе), то перестает передавать трафик группового вещания для этой группы. Хост может также быть исключен из группы, просто не отвечая маршрутизатору на запрос о членстве (такой подход реализован в протоколе IGMPv1). Тогда маршрутизатор будет продолжать передавать нежелательный трафик группового вещания, пока не истечет некоторый период времени с момента поступления последнего отчета о членстве. Такой подход значительно удлиняет период скрытого нахождения хоста в состоянии выхода из группы, что снижает эффективность работы сети.

Локальная сеть может иметь несколько хостов, заинтересованных в получении трафика одной и той же группы, но маршрутизатору достаточно подтверждения только от одного хоста, чтобы продолжать передачу трафика в сеть для этой группы. При использовании протокола IGMPv2 для ограничения числа ответов хостов на запрос маршрутизатора любой хост, состоящий в группе, вместо того чтобы немедленно ответить на запрос, сначала некоторое, определенное протоколом время ждет, не появится ли в сети ответ какого-нибудь другого хоста. Если по истечении этого времени ответ другого хоста не поступил, то он посылает маршрутизатору собственный отчет о членстве. (Если же используется протокол IGMPv3, то никаких пауз не устанавливается и хосты сразу генерируют сообщения о членстве.)

ПРИМЕЧАНИЕ

Чтобы хост смог получать трафик группового вещания, недостаточно установить на нем протокол IGMP, с помощью которого хост может отправить сообщение своему маршрутизатору о желании присоединиться к группе. Помимо этого, надо сконфигурировать сетевой интерфейс хоста так, чтобы он стал захватывать из локальной сети кадры, несущие в себе пакеты группового вещания для той группы, к которой присоединился хост. Для этого необходимо настроить интерфейс на прослушивание определенного группового адреса канального уровня, соответствующего групповому IP-адресу. К сожалению, адресное пространство групповых IP-адресов в 32 раза объемнее пространства групповых MAC-адресов. То есть отображение этих двух адресных пространств друг на друга оказывается далеко не однозначным — на один и тот же групповой MAC-адрес отображается целый блок из 32 различных групповых IP-адресов. Следовательно, когда сетевой адаптер захватывает кадр, содержащий пакет группового вещания, существует значительная вероятность того, что этот пакет был направлен совсем другой группе. Однако эта ошибка скоро обнаруживается. Когда кадр передается вверх по стеку, протокол IP проверяет, совпадает ли групповой IP-адрес в поле адреса назначения инкапсулированного пакета с групповым IP-адресом данного интерфейса (при этом ни групповые IP-адреса, ни групповые MAC-адреса никогда не используются в качестве адресов отправителя).

Принципы маршрутизации трафика группового вещания

Среди принципов маршрутизации трафика группового вещания можно отметить:

- маршрутизацию на основе доменов;
- учет плотности получателей группового трафика;
- два подхода к построению маршрутного дерева;
- концепцию продвижения по реверсивному пути.

Маршрутизация на основе доменов. Значительный объем хранимой и передаваемой по сети служебной информации, используемой для поддержания группового вещания, стал фактором, ограничивающим масштабируемость данной технологии. Для улучшения масштабируемости разработчики технологии группового вещания предложили традиционный для интернета иерархический подход, основанный на доменах. Подобно автономным системам (доменам маршрутизации) и DNS-доменам вводятся **домены группового вещания**. Для доставки информации в пределах домена предлагаются одни методы и протоколы маршрутизации группового вещания, называемые *внутридоменными*, а в пределах многодоменной структуры — другие, называемые *междоменными*. Мы ограничимся здесь описанием свойств внутридоменных средств продвижения пакетов группового вещания.

Учет плотности получателей группового трафика. Внутридоменные протоколы маршрутизации разделяются на два принципиально отличных класса:

- Протоколы **плотного режима** (Dense Mode, DM) разработаны в предположении, что в сетевом домене существует большое число принимающих узлов. Отсюда следует главная идея этих протоколов: сначала затопить сеть пакетами группового вещания по всем направлениям, останавливая продвижение пакетов, лишь когда находящийся на пути распространения трафика маршрутизатор явно сообщит, что далее ниже по потоку членов данной группы нет.
- Протоколы **разряженного режима** (Sparse Mode, SM) рассчитаны на работу в сети, в которой количество маршрутизаторов с подключенными к ним членами групп невелико по сравнению с общим числом маршрутизаторов. В этой ситуации выгоднее не усекать некоторые пути распространения широковещательной рассылки, а использовать явные сообщения о необходимости присоединения подсетей к дереву рассылки.

В сети, использующей протокол класса SM, необходимо существование центрального элемента, обычно называемого **точкой рандеву** или **встречи** (rendezvous point). Точка встречи должна существовать для каждой имеющейся в сети группы и быть единственной для группы. Все узлы, заинтересованные в получении информации, предназначенной той или иной группе, должны регистрироваться в соответствующей точке встречи. Функции точки (или нескольких точек) встречи выполняет специально назначенный для этого маршрутизатор. В сети может быть несколько маршрутизаторов, играющих роли точек встречи.

Два подхода к построению маршрутного дерева. Как и при решении задачи маршрутизации на основе индивидуальных адресов, в сети с групповым вещанием маршрутизаторы анализируют топологию сети, пытаясь найти кратчайшие пути доставки данных от источников к получателям. При этом все протоколы маршрутизации группового вещания используют один из следующих двух подходов:

- Для всех источников данной группы строится *единственный* граф связей, называемый **разделяемым деревом**. Этот граф связывает всех членов данной группы (точнее, все маршрутизаторы, к которым подключены локальные сети, имеющие в своем составе членов данной группы). Разделяемое дерево может включать также и необходимые для обеспечения связности маршрутизаторы, не имеющие в своих присоединенных сетях членов данной группы. Разделяемое дерево служит для доставки трафика всем членам данной группы от *каждого* из источников, вещающих на данную группу.
- Для каждой группы строятся *несколько* графов по числу источников, вещающих на каждую из этих групп. Каждый такой граф, называемый **деревом с вершиной в источнике**, служит для доставки трафика всем членам группы, но только от *одного* источника.

Концепция продвижения по реверсивному пути. Механизм, используемый для маршрутизации трафика группового вещания, в определенном аспекте является прямо противоположным (реверсивным) традиционному способу маршрутизации на основе индивидуальных адресов, при котором маршрутизаторы перемещают пакет по сети в направлении приемника. Напротив, все пакеты с групповым адресом маршрутизаторы тиражируют и передают копии во все стороны — на все интерфейсы, кроме того, с которого этот пакет поступил.

При этом в сложных сетях возможно образование петель. Для правильной работы сети зациклившиеся пакеты необходимо распознавать и отбрасывать. Петля не может возникнуть, если пакет прибыл от источника по ожидаемому пути, проложенному в соответствии с обычным алгоритмом маршрутизации, основанном на анализе таблиц маршрутизации.

Маршрутизатор проверяет, является ли входной интерфейс, получивший групповой пакет, интерфейсом, через который пролегает кратчайший путь к источнику, с помощью обычной таблицы маршрутизации, которая, как известно, содержит указания о рациональных путях ко всем сетям составной интерсети. Проверка факта выполнения данного условия называется **продвижением по реверсивному пути** (Reverse Path Forwarding, **RPF**). Название объясняется тем, что эта процедура связана не столько с путями, ведущими вперед от текущего места нахождения пакета к пункту назначения, сколько с обратным (реверсивным) путем, который уже пройден пакетом от того места, где он находится сейчас, до источника. Только пакеты, прошедшие RPF-проверку, являются кандидатами для дальнейшего продвижения вдоль путей, ведущих к потенциальным получателям трафика группового вещания.

Концепция продвижения по реверсивному пути является главной при маршрутизации группового трафика независимо от того, какой протокол при этом использован. Механизм RPF применятся и в других вариантах организации группового вещания. Например, когда маршрутизатор пытается продвигать пакеты к точке встречи в сети, работающей в разряженном режиме, он выбирает интерфейс, от которого проходит кратчайший путь к точке встречи.

(S) *Протоколы маршрутизации группового вещания*

(S) *Междоменное групповое вещание*

Программно-определяемые сети SDN

Недостатки традиционной модели маршрутизации

Классические сетевые устройства — коммутаторы и маршрутизаторы — *весьма негибкие устройства,* когда дело касается продвижения кадров и пакетов. Хотя их основу и составляет специализированный компьютер с развитым программным обеспечением, которое в принципе позволяет реализовать сложные адаптивные алгоритмы обработки и продвижения трафика, на практике основным способом продвижения пакетов является продвижение на основе значения единственного поля заголовка пакета — адреса назначения. Для изменения этого способа администратору сети приходится затрачивать немалые усилия по конфигурированию сетевых устройств. Например, для того чтобы маршрутизатор начал принимать во внимание при продвижении пакета не только IP-адрес назначения, но и IP-адрес источника, администратор должен внести новые строки в файл конфигурации[1].

Для придания гибкости в процессе продвижения трафика возможна маршрутизация на основе правил (Policy-Based Routing, PBR). Мы видели пример такого подхода в протоколе BGP. Но реализация такой идеи в IGP-маршрутизаторах приводит к существенному замедлению процесса продвижения пакетов, так как в отличие от протокола BGP, внутренние протоколы маршрутизации разрабатывались без учета такой функциональности и отрабатывают правила маршрутизации не в специализированных микросхемах, как это происходит при стандартной OSPF/IS-IS/RIP маршрутизации, а программным способом, в отдельном приложении из сетевой операционной системы.

Децентрализованные протоколы построения таблиц маршрутизации *медленно реагируют на изменения топологии* сети из-за того, что информация об этих изменениях распростра-

[1] См. раздел «Фильтрация» главы 30.

няется по цепочке, от одного маршрутизатора к другому, а не направляется сразу в единый центр принятия решений. В результате в некоторые периоды времени маршрутизаторы, обладая неверной информацией о топологии сети, принимают ошибочные решения о продвижении пакетов.

На интернет-магистралях, где маршрутизаторы передают агрегированные потоки, состоящие из сотни тысяч индивидуальных потоков и главной задачей является их доставка в соответствующую периферийную сеть без какой-либо дополнительной обработки, не столь важны гибкость и адаптивность сети. Но если, например, взглянуть на центр обработки данных облачного провайдера, то здесь *различные услуги требуют различной обработки трафика*: услуги виртуальных частных сетей, виртуальных файерволов или виртуальных офисов, динамическое перераспределение нагрузки между серверами центра данных — все это требует логического разделения трафика на основе специальной фильтрации на отдельные потоки и различного продвижения каждого потока, что представляет собой сложную задачу для негибких традиционных коммутаторов и маршрутизаторов.

Еще одним недостатком традиционных коммутаторов и маршрутизаторов является их *нестандартный командный язык конфигурирования* устройств. Это приводит к тому, что централизованной системе управления сетью необходимо иметь различные программные подсистемы для управления устройствами различных производителей

Концепция **программно-определяемых сетей** (Software Defined Networks, **SDN**) призвана устранить недостатки традиционной модели маршрутизации.

Принципы организации сетей SDN

В самом общем виде под концепцией программно-определяемых сетей понимается *разделение функций управления и функций продвижения* сетевого устройства на основе некоторой модели (абстракции) сетевого устройства и передача функций управления программным приложениям (рис. 16.9). Эти приложения работают в отдельном центральном узле сети (контроллере SDN), непосредственно получают информацию от сетевых устройств в реальном времени и могут управлять устройствами динамически, а следовательно, и управлять поведением сети в целом.

Интерфейс между приложениями и сетевыми устройствами стандартизован. Этот интерфейс основан на единой модели (абстракции) функциональности слоя продвижения сетевого устройства.

Управление функциями продвижения сетевых устройств с помощью внешних приложений делает поведение сети гибким за счет его программируемости. Жесткое правило продвижения на основе только адреса назначения отменяется и заменяется алгоритмами, содержащимися в приложениях, работающими на контроллере SDN.

Существуют различные реализации концепции SDN. Они отличаются степенью перераспределением функций слоя управления между устройством и контроллером.

На одном краю спектра, который считается классическим, потому что реализует идею SDN в наиболее последовательном, можно сказать, в чистом виде, находятся *сети SDN на основе протоколов OpenFlow и P4*. В этом типе сетей сетевое устройство максимально упрощено, практически лишено функции слоя управления и только способно отрабатывать простые инструкции по продвижению кадров, передаваемые устройству контроллером. Эту схему

разделения функций слоя продвижения и управления можно назвать «“глупое” устройство — “умный” контроллер».

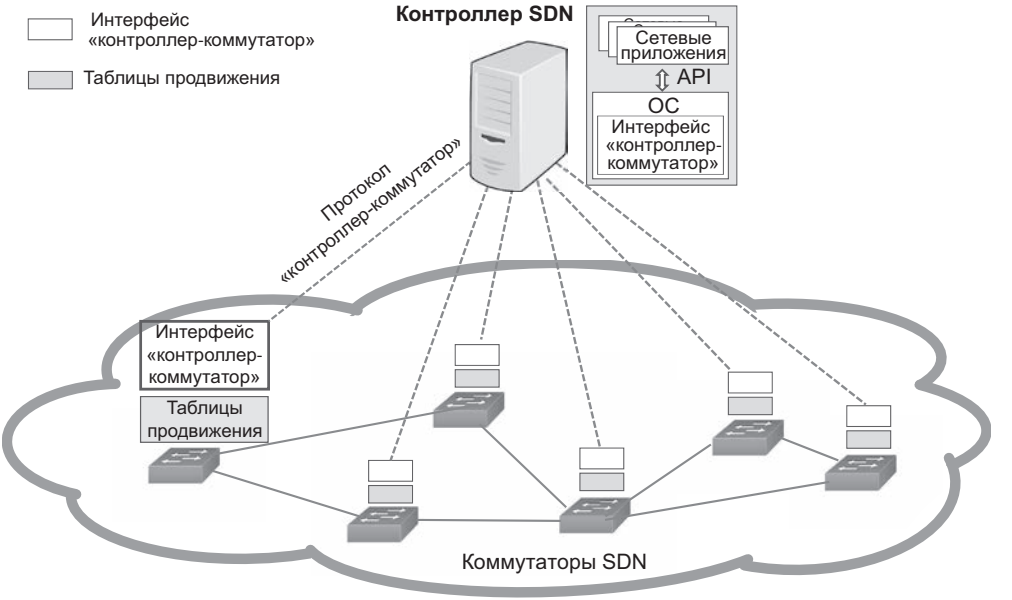


Рис. 16.9. Архитектура сети SDN OpenFlow

На другом краю спектра находятся *сети SDN, в которых сетевые устройства остаются «умными»*: они имеют развитую функциональность слоя управления, поддерживают протоколы распределенной маршрутизации, функции виртуальных локальных сетей, файерволов и некоторые другие, и при этом поддерживают некоторые расширения этих протоколов, позволяющие им принимать инструкции от контроллера SDN, которые программируют их поведение. Эти инструкции не определяют пошаговое поведение устройства при продвижении пакета, как это происходит в системах на основе протоколов OpenFlow/P4, а содержат высокоуровневые намерения (intents) приложения, работающего на контроллере SDN. Таким намерением может быть инструкция по созданию новой сети VLAN и соединения с ее помощью нескольких хостов, создание файервола между несколькими IP-подсетями, организация туннеля с шифрацией трафика между подсетями. По сути, контроллер SDN в этом варианте создает услуги с помощью «умных» сетевых устройств, которым достаточно сообщить о намерении, а они способны реализовать эти намерения самостоятельно.

Между этими двумя полюсами находятся различные системы SDN, в которых контроллер программирует одну вполне определенную функцию продвижения пакетов сетевым устройством, не претендуя на создание разнообразных услуг. Примером таких систем является система сегментной маршрутизации трафика в сетях SR-MPLS, в которой контроллер работает на основе протокола PCEP или BGP. Принципы работы этой системы рассмотрены в главе 20 в разделе «Инжиниринг трафика в MPLS».

Далее мы изучим архитектуру системы SDN на основе протоколов OpenFlow и P4.

Сети SDN на основе протокола OpenFlow

Модель коммутатора и протокол OpenFlow (OF) были предложены в 2006 году исследователями Стэнфордского университета и Университета Беркли. Именно эта работа и положила начало утверждению концепции программно-управляемых сетей SDN, хотя идея разделения слоя управления и слоя продвижения данных обсуждалась и применялась и ранее. Например, в стационарных цифровых телефонных сетях 80-х годов применялись протоколы сигнализации SS7, образовавшие отдельную от сети передачи голосовых данных сеть.

Изначально протокол и коммутатор OpenFlow создавались как средство быстрой разработки и тестирования новых сетевых протоколов, то есть как инструмент исследователей. Однако довольно быстро пришло понимание, что его гибкость и мощность могут быть использованы для изменения к лучшему функциональности сетей.

Модель коммутатора OpenFlow использует **таблицу продвижения**, состоящую из ряда записей — **правил обработки** пакетов. В исходном состоянии таблица продвижения коммутатора пуста. Ее формирование — это обязанность приложений контроллера SDN. Полученные от приложений правила обработки пакетов контроллер передает коммутатору по протоколу OF. Помимо сообщений-*правил* в число возможных сообщений протокола OF входят также сообщения-*запросы*, с помощью которых контроллер запрашивает у коммутатора информацию о состоянии его портов (работоспособные или нет), а также статистику потоков. В протоколе OF предполагается, что коммутатор не только отвечает на запросы контроллера, но может передать контроллеру сообщения по *своей инициативе*, например, в случае изменения состояния порта или удаления некоторого правила по тайм-ауту. Канал обмена сообщениями между контроллером и коммутатором SDN называется **управляющим каналом**. Он представляет собой *TCP-сессию*, установленную в IP-сети между контроллером и коммутатором (здесь можно упрекнуть создателей протокола OpenFlow в нарушении соблюдения принципа «"умный" контроллер — "глупое" устройство», так как для поддержания TCP-сессии устройство должно поддерживать протоколы IP и TCP).

Каждое **правило** может включать элементы трех типов:

- *Условия* выделения потока пакетов, к которым это правило должно быть применено.
- *Действия*, которые должны быть выполнены над пакетом, который удовлетворяет условиям данного правила.
- *Счетчики*, измеряющие характеристики потока пакетов.

Условия строятся на основе значений полей заголовка пакета, а также номера порта коммутатора, на который поступил кадр.

Примером условия может быть такой набор значений:

Conditions:

{Port = 5; VLAN ID = 254; IP Src = 194.81.18.227; IP Dst = 130.16.55.12; TCP Dst Port = 5009}

Это условие определяет поток пакетов, поступающих на порт 5, относящийся к VLAN 254, отправленных хостом с IP-адресом 194.81.18.227 хосту с IP-адресом 130.16.55.12 на TCP-порт 5901. Условие считается выполненным, если выполняются все его компоненты, то есть они объединяются логическим «И». Как видно из примера, в условии не обязательно использовать все возможные поля заголовка кадра; те поля, которые в условии не указаны, считаются замаскированными, то есть они могут иметь любые значения. Поля IP-адресов

могут иметь маску переменной длины, задающую значимую часть адреса, то есть возможно задать префикс 130.16.55.0/24 вместо адреса хоста 130.16.55.12.

Конструкция условий протокола OF позволяет очень тонко выделить поток данных и тем самым обеспечить его индивидуальную обработку коммутатором. Сравните это с возможностями стандартных коммутаторов и маршрутизаторов, которые считают одним потоком все пакеты, у которых совпадает лишь один признак — адрес назначения.

Набор возможных **действий** некоторого правила включает в себя передачу пакета на один из выходных портов коммутатора и операции по изменению значений полей заголовка пакета. Например, для того чтобы пакеты потока были переданы на порт 8 и чтобы при этом номер VLAN был изменен на 1002, а MAC-адрес назначения — на bc:30:7e:d4:5d:8f, достаточно определить набор из трех действий:

Actions:
{Port=8; VLAN ID = 1002, MAC Dst = bc:30:7e:d4:5d:8f}

Счетчики позволяют коммутатору измерять статистические характеристики потока: его интенсивность (количество поступивших пакетов за секунду) и продолжительность существования. Коммутатор также должен поддерживать агрегатные счетчики для каждого порта. Счетчики обновляются при поступлении каждого нового пакета в коммутатор, но непосредственно на обработку пакетов не влияют. Контроллер SDN может запросить у коммутатора значения счетчиков и на основании этих данных принять решение, например, заблокировать некоторый подозрительный поток, если его интенсивность слишком велика.

Для принятия решения контроллеру нужно знать топологию сети и для этой цели был разработан протокол **BDDP** (Broadcast Domain Discovery Protocol), с помощью которого устройства сети обнаруживают соседей и сообщают об этом контроллеру.

Помещение всех правил, учитывающих всевозможные сочетания условий, в одну таблицу не является масштабируемым решением, поэтому устройства OpenFlow поддерживают конвейер таблиц, в который может входить до 255 таблиц. При обработке пакета устройство последовательно просматривает все таблицы конвейера, если только в одной из них не содержится правило выхода из конвейера.

Таблицы не хранят информацию о состоянии потоков данных (кроме специальной таблицы счетчиков, которые подсчитывают скорость потока для его динамического профилирования). Состояние потока может переноситься самими пакетами потока в специальном дополнительном поле заголовка — поле метаданных.

У протокола OpenFlow имеются несколько проблем, препятствующих его широкому применению. Одной из главных является проблема унификации конвейера таблиц. Другими словами, проблемой является создание единой модели сетевого устройства, которой бы следовали все производители сетевого оборудования, желающие выпускать оборудование SDN.

Решением этой проблемы явилось создание программируемых протокольно-независимых процессоров P4, которые позволяют создавать нужную модель сетевого устройства за счет программирования процессора этого устройства. Таким образом, вместо нахождения единой унифицированной модели коммутатора предлагается коммутатор, характеристики которого можно изменить по желанию проектировщика сети, а потом уже управлять им с помощью контроллера SDN.

Программируемые протокольно-независимые пакетные процессоры (P4)

Прогресс в области производства микросхем за 8 лет, прошедшие с момента зарождения технологии OF SDN, привел к тому, что в 2014 году была высказана идея построения коммутатора на сравнительно недорогих программируемых микросхемах, в которых функции не являются фиксированными, так как реализуются процессорами.

Основу коммутатора технологии P4 составляет программируемый пакетный процессор — микросхема, функциональные возможности которой программируются на предварительном *этапе конфигурирования* коммутатора. На этом этапе программист задает, какие поля заголовка пакета должны приниматься во внимание в правилах продвижения, какой набор условий и действий допускается применять в правилах продвижения, а также какой набор таблиц может поддерживать конвейер таблиц коммутатора. Упрощенно можно сказать, что программист может задать конфигурацию OF-коммутатора определенной модели компании Cisco, так как эта конфигурация в полной мере подходит требованиям приложения A, для приложения B — определить конфигурацию модели коммутатора HP, а для приложения C — создать уникальную конфигурацию, не встречавшуюся в моделях OF-коммутаторов. После определения архитектуры коммутатора P4 наступает второй этап, который соответствует *этапу загрузки правил* в OF-коммутатор. На этом этапе приложение, работающее на контроллере, создает и загружает в таблицы коммутатора P4 набор правил, определяющих способ обработки и продвижения пакетов при поступлении их на входные порты коммутатора. И наконец, на третьем *этапе управления* коммутатор P4 начинает обрабатывать пакеты в соответствии с загруженными в него правилами. Этапы работы коммутатора P4 иллюстрирует рис. 16.10.

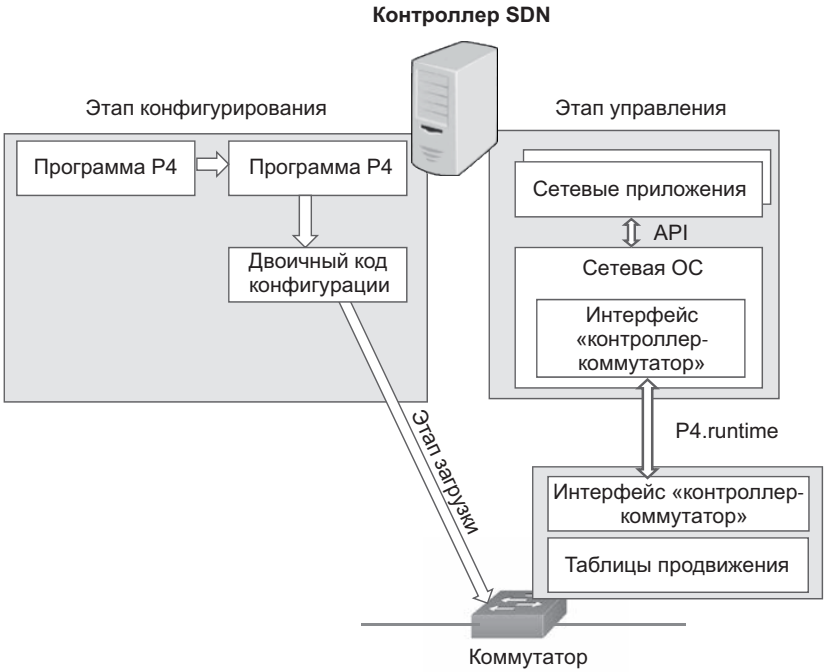


Рис. 16.10. Программное конфигурирование коммутатора P4

На этапе конфигурирования разрабатывается **программа** P4, которую **компилятор** P4 транслирует в двоичный код конфигурации. Этот двоичный код загружается в коммутатор, после чего он готов принимать правила продвижения пакетов и выполнять их в реальном времени.

Для реализации второго и третьего этапов можно было бы использовать уже существующий протокол OpenFlow, но архитекторы P4 решили заменить его новым **протоколом p4.runtime**, так как в сообщениях OpenFlow за типами заголовков и действий жестко закреплены определенные числовые значения и это делает его зависимым от конфигурации коммутатора. Программируемость архитектуры контроллера P4 решает проблему унификации коммутаторов OF: вместо описания всеобъемлющей архитектуры, которая должна включать все возможные варианты структуры заголовка пакета и все возможные условия и действия над пакетами, создается возможность запрограммировать желательную для определенного класса приложений архитектуру коммутатора.

Контроллеры P4 находят применение в сетях центров данных, где проблема масштабируемости централизованного решения не стоит остро и может быть решена на основе кластера контроллеров. Подробное описание архитектуры SDN на основе OpenFlow и P4 можно найти на сайте книги.

(S) *Технологии OpenFlow и P4*

Виртуализация сетевых функций: NFV

Еще одним новым направлением обеспечения программируемости сетей наряду с рассмотренной концепцией программно-определяемых сетей SDN является концепция **виртуализации сетевых функций** (Network Function Virtualisation, **NFV**).

Эта концепция состоит в том, что *функции сетевых устройств* — маршрутизаторов, коммутаторов, файерволов и др. — реализуются не на основе аппаратной специализированной платформы, а *программным путем в серверах общего назначения*.

Можно считать подход NFV развитием популярной концепции виртуальных машин (VM), согласно которой, напомним, за счет слоя виртуализации — гипервизора — на основе физических ресурсов одного компьютера создается несколько виртуальных машин со своей собственной ОС и приложениями. В виртуальном мире NFV создаются программные единицы, эмулирующие не только физический компьютер, но и физические сетевые устройства, в результате чего программным путем создается **виртуальная сеть**, функционально аналогичная реальной сети.

Идею NFV иллюстрирует рис. 16.11. На нем мы видим три физических сервера, работающих в центре данных некоторого провайдера услуг. Серверы соединены с внешним миром и между собой физическими связями, показанными на рисунке сплошными линиями. Эти физические связи образованы коммутаторами и маршрутизаторами центра данных. На рисунке они не показаны, так как их характеристики и структура связей не важны для нас. Главное, что они позволяют серверам взаимодействовать друг с другом и с внешним миром и создавать программным путем виртуальные сети для предоставления услуг пользователям.

Для того чтобы провайдер мог сконфигурировать виртуальную сеть, пользователь должен описать функции этой сети. Описание может быть как формальным, то есть составлен-

ным на некотором специальном языке описания алгоритмов обработки входных потоков данных, так и неформальным, когда пользователь описывает примерную структуру сети и ее функции, используя привычные ему блоки — маршрутизаторы и коммутаторы. В это описание должно также включаться описание возможных входных потоков для виртуальной сети и правил их обработки.

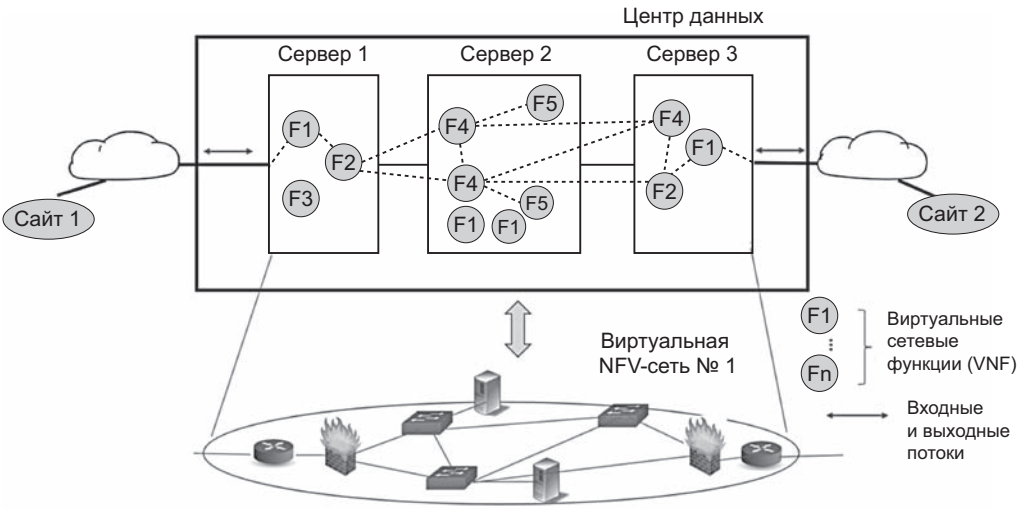


Рис. 16.11. Виртуальная NFV-сеть N1

Рассмотрим пример такого неформального описания. Пусть пользователю необходимо иметь у провайдера два сервера приложений, на которых будут работать почтовые серверы и веб-серверы его предприятия. Пользователь хотел бы, чтобы провайдер обеспечил к этим серверам доступ сотрудников предприятия, базирующихся на двух сайтах, которые будут соединены с центром данных провайдера через интернет. К веб-серверам также должны иметь доступ все пользователи интернета, но при этом их права должны быть ограничены просмотром только определенных страниц. Результатом обсуждения требований пользователя с провайдером стала виртуальная сеть N1, состоящая из элементов, выполняющих функции традиционных коммутаторов и маршрутизаторов, защищенная двумя файерволами (показана в нижней части рис. 16.11). Нужно подчеркнуть, что на этом этапе речь идет лишь об описании функций сети в терминах традиционных сетевых элементов — провайдеру еще предстоит реализовать эту сеть, пользуясь техникой NFV. Для реализации этой сети в каждом сервере провайдера имеется набор программных модулей, каждый из которых эмулирует некоторую сетевую функцию. Говорят, что такой элемент является **виртуальной сетевой функцией** (Virtual Network Function, **VNF**). У серверов провайдера имеется библиотека программных модулей VNF пяти типов:

- F1 — IP-маршрутизатор;
- F2 — файервол (о его функциях см. главу 30, но для нашего примера достаточно знать, что это устройство блокирует трафик внешних атак на сеть);
- F3 — виртуальная машина под управлением ОС Ubuntu;

- F4 — коммутатор Ethernet;
- F5 — виртуальная машина под управлением ОС Centos.

Для создания требуемой сети в первом сервере активизированы экземпляры функций F1 и F2, при этом между ними создается программная связь, по которой между ними будут передаваться данные. Во втором сервере активизированы два экземпляра функции коммутатора Ethernet F4 и два экземпляра функции виртуальной машины под управлением ОС Centos, соединенные программными связями в соответствии с топологией создаваемой сети N1.

Аналогичная работа выполнена и в третьем сервере. В результате виртуальная сеть N1 приобрела законченный вид и готова обрабатывать реальные пакеты реального пользователя, приходящие от его сайтов, а также трафик внешних пользователей, то есть оказывать вполне определенную услугу.

На рисунке также показано, что на первом и втором сервере провайдера активизированы функции, которые не принадлежат создаваемой виртуальной сети N1. Это сделано для того, чтобы подчеркнуть тот факт, что на серверах провайдера одновременно сосуществуют виртуальные сети многих пользователей: провайдер может создавать их программно, не изменяя нижележащую аппаратную инфраструктуру.

Заметим, нами пока использован привычный для специалиста по компьютерным сетям термин «виртуальная сеть» для описания результата связывания виртуальных функций друг с другом. В «новом мире» NFV принято говорить не о виртуальной сети, а о виртуальной сетевой услуге, которую оказывает **цепочка функций VNF** (VNF chain). Сам процесс связывания сетевых функций для образования законченной услуги называется **оркестрированием** (orchestration). Основными преимуществами виртуализации сетевых функций считаются:

- Независимость от индивидуальной архитектуры традиционных сетевых устройств конкретного производителя, то есть возможность унификации функций и интерфейсов менеджмента сетевых устройств в их программной реализации. Как видим, это та же цель, которую преследуют и создатели концепции программно-определяемых сетей SDN, но достигается эта цель другими средствами.
- Снижение стоимости сетевого оборудования, так как стандартные серверы обычно стоят дешевле специализированных аппаратных устройств — коммутаторов, маршрутизаторов и др. (со сравнимой производительностью).
- Повышение динамичности и простоты создания новых сетевых услуг провайдерами, так как новая услуга создается программным путем за счет связывания необходимых функций в нужную цепочку.
- Простота создания новых сетевых функций, отвечающих требованиям новых сетевых приложений, — запрограммировать новую функцию в среде Linux легче, чем реализовать ее на базе жесткой аппаратной платформы конкретного производителя.

В то же время очевидны и недостатки подхода NFV:

- Производительность программной реализации сетевой функции на базе универсального сервера может оказаться недостаточной, значительно уступающей производительности специализированного устройства, такого как аппаратный маршрутизатор или коммутатор. Для обеспечения нужной производительности сетевые функции должны

выполняться на кластерах серверов, возможно, распределенных между несколькими центрами данных.

- Надежность также может быть проблемой, поскольку специализированная аппаратура обычно имеет более высокую надежность, чем универсальные серверы.
- Автоматизация процесса создания библиотек сетевых функций и оркестрирования их в сетевую услугу требует разработки новых стандартов и новых программных средств.

Организация, которая первой в 2012 году сформулировала идею NFV, — Европейский институт телекоммуникационных стандартов ETSI — предложила и стандартную архитектуру, которая описывает основные компоненты автоматизированной программной системы, поддерживающей жизненный цикл создания услуг на основе сетевых функций VNF. Эта архитектура получила название NFV **MANO** — от NFV **MAN**agement and **O**rchestration, то есть менеджмент и оркестрирование NFV.

Разработчики преследовали три основные цели:

- полное разделение аппаратных и программных средств;
- автоматическое и масштабируемое применение сетевых функций VNF;
- детальный контроль состояния сети и сетевых функций.

Три менеджера этой архитектуры контролируют различные слои сети. Менеджер виртуализированной инфраструктуры сети контролирует нижний уровень — физические ресурсы сети и их отображение в виртуальные машины и виртуальные хранилища данных. Менеджер виртуальных функций контролирует процесс активизации и работы отдельных виртуальных функций VNF. Менеджер верхнего уровня иерархии, или NFV-оркестратор, координирует процесс создания цепочек виртуальных функций для предоставления определенной услуги сети.

Архитектура NFV MANO стандартизует интерфейсы между ее менеджерами и объектами, которыми они управляют, что создает основу для разработки совместимых элементов этой архитектуры разными производителями и группами разработчиков. Институт ETSI занимается также разработкой языка формального описания сетевых функций и услуг на их основе.

Существуют различные программные реализации функций NFV и менеджеров архитектуры MANO, в том числе Open Source MANO (OSM) от ETSI.

Технологии NFV и SDN служат одной и той же цели — повышению гибкости и адаптивности сети за счет программируемости. Эти технологии различны, но они хорошо дополняют друг друга. Они могут применяться независимо друг от друга, но могут и сосуществовать в одной сети. Так, коммутатор SDN может служить одним из физических ресурсов сети, в этом случае контроллер SDN будет работать как отдельный элемент менеджера виртуализированной структуры сети VIM. Коммутатор SDN может также быть реализован как сетевая функция VNF — в этом случае контроллер SDN будет одним из элементов менеджера виртуальных функций.

(S) *Функциональная схема и классификация маршрутизаторов. Операционная система Cisco/OS*

ГЛАВА 17 IPv6 как развитие стека TCP/IP

Исторические предпосылки

В начале 90-х стек протоколов TCP/IP столкнулся с серьезными проблемами. Именно тогда началось активное промышленное использование интернета: переход к построению сетей предприятий на основе интернет-транспорта, применение веб-технологии для доступа к корпоративной информации, ведение электронной коммерции через интернет, внедрение интернета в индустрию развлечений (распространение видеофильмов, звукозаписей, интерактивные игры). Все это привело к резкому росту числа узлов сети, изменению характера трафика и ужесточению требований, предъявляемых к качеству обслуживания сетью ее пользователей. Сообщество интернета, а вслед за ним и весь телекоммуникационный мир, начали решать новые задачи путем создания новых протоколов для стека TCP/IP, таких как протокол резервирования ресурсов RSVP, защищенный протокол IPSec, протокол коммутации меток MPLS и т. п. Однако ведущим специалистам было ясно, что только за счет добавления новых протоколов технологию TCP/IP развивать нельзя — нужно решиться на *модернизацию сердцевины стека*, протокола IP. Некоторые проблемы нельзя было решить без изменения формата IP-пакета и логики обработки полей заголовка IP-пакетов. Наиболее очевидной проблемой такого рода была проблема дефицита IP-адресов, которую невозможно снять, не расширив размер полей адресов источника и приемника.

Наряду с добавлением новых функций непосредственно в протокол IP необходимо было обеспечить его тесное взаимодействие с *новыми протоколами* стека TCP/IP, что также требовало добавления в заголовок IP новых полей, обработку которых осуществляли бы эти протоколы. Например, для работы RSVP было желательно введение в заголовок IP поля метки потока, а для протокола IPSec — специальных полей для передачи данных, поддерживающих функции обеспечения безопасности. После достаточно долгого обсуждения интернет-сообщество решило подвергнуть протокол IP серьезной переработке. В качестве основных целей модернизации были выбраны следующие:

- создание масштабируемой схемы адресации;
- развитие способности сети к автоконфигурированию;
- сокращение объема работы, выполняемой маршрутизаторами;
- предоставление гарантий качества транспортных услуг;
- обеспечение защиты данных, передаваемых по сети.

В августе 1998 года были приняты регламентирующие документы, определяющие как общую архитектуру новой, **шестой версии протокола IPv6** (RFC 8200), так и его отдельные аспекты, например систему адресации (RFC 4291).

Система адресации IPv6

Отличие от IPv4

Система адресации IPv6 внесла существенные изменения в прежнюю систему. Прежде всего увеличилась разрядность адреса: вместо 4 байт IP-адреса в версии IPv4 в новой версии под адрес отведено *16 байт*. Это дает возможность пронумеровать огромное количество узлов:

340 282 366 920 938 463 463 374 607 431 762 211 456.

Масштаб этого числа иллюстрирует, например, такой факт: если разделить это теоретически возможное количество IP-адресов между всеми жителями Земли (а их сегодня примерно 6 миллиардов), то на каждого из них придется невообразимо, если не сказать бессмысленно, большое количество IP-адресов — $5{,}7 \times 10^{28}$! Очевидно, что такое значительное увеличение длины адреса было сделано не только и даже не столько для снятия проблемы дефицита адресов.

> Главной целью изменения системы адресации было не механическое увеличение адресного пространства, а повышение эффективности работы стека TCP/IP в целом.

В новой версии не поддерживаются классы адресов (A, B, C, D, E), но широко используется технология CIDR. Благодаря этому, а также усовершенствованной системе групповой адресации и введению адресов нового типа IPv6 может *сократить затраты на маршрутизацию*. Произошли и чисто внешние изменения — разработчики стандарта предложили использовать вместо десятичной *шестнадцатеричную* форму записи IP-адреса. Каждые четыре шестнадцатеричные цифры отделяются друг от друга двоеточием. Например, адрес IPv6 может выглядеть так: FEDC:0A98:0:0:0:0:7654:3210. Старшие нули в группе четырех шестнадцатеричных цифр могут быть опущены; так, вместо 0A98 можно писать A98. Несколько нулевых групп могут быть заменены парой двоеточий — FEDC:0A98::7654:3210. В адресе допускается только одна такая замена. Для сетей, поддерживающих обе версии протокола (IPv4 и IPv6), разрешается задействовать для младших четырех байтов традиционную для IPv4 десятичную запись: 0:0:0:0:0:FFFF:129.144.52.38.

В IPv6 *расширился набор адресов* разного типа, приписываемых одному сетевому интерфейсу, каждый из которых используется для специфических целей. Для IPv4 является обычной ситуацией, когда хост, имеющий один сетевой адаптер, получает один IP-адрес. В IPv6 каждый интерфейс всегда получает несколько IP-адресов. Например, всем интерфейсам в сети IPv6 автоматически приписываются адреса, которые они могут использовать для обмена локальным трафиком, а для глобального трафика назначаются дополнительные адреса, подобные публичным индивидуальным адресам IPv4. Помимо стандартных (well known) групповых адресов, каждый интерфейс имеет особый групповой адрес, идентифицирующий группу, состоящую только из него самого. Эти, а также другие типы адресов интерфейса направлены на достижение конкретных целей: уменьшение вспомогательного трафика, отказ от широковещательной рассылки (в том числе и на канальном уровне), поддержка автоконфигурации и др.

Важным усовершенствованием в системе адресации IPv6 стало введение для каждого адреса признака **области действия адресов** (scope), определяющего ту часть сети, в пределах

которой этот адрес является *действительным* и *уникальным*. Так, пакет с адресом уровня линии связи[1] (link-local) может использоваться в коммуникациях смежных узлов, но будет отвергнут первым же маршрутизатором на пути в другой сегмент сети, так как область действия этого адреса ограничена линией связи. Как известно, в IPv4 также имеются адреса, область действия которых различается, — публичные адреса, используемые в сетях, подсоединенных к интернету, и частные адреса (сетей 10.0.0.0 и др.) для внутренних целей. Если пакет с частным адресом поступит (по ошибке) на внешний маршрутизатор, то он в общем случае может быть передан дальше в публичную сеть.

В IPv6 дело обстоит по-другому. Маршрутизаторы и хосты в сети IPv6 рассматривают каждый адрес в совокупности с определенной для него областью действия. При поступлении на маршрутизатор пакета с адресом, область действия которого ограничена внутренней сетью предприятия, маршрутизатор не пропустит его в интернет, где действительны адреса с глобальным признаком области действия. Хосты также принимают во внимание область действия адресов. Хост, формируя пакет, учитывает области действия адресов. Например, он не должен отправлять пакет, в котором адрес назначения глобальный, а адрес источника — адрес уровня линии связи. Поскольку признак scope определяет ту часть сети, в которой тот или иной адрес должен быть уникальным, то один и тот же индивидуальный адрес может использоваться разными хостами в разных частях сети.

И наконец, одним из наиболее полезных свойств технологии IPv6 является развитая способность сетевых устройств IPv6 к **автоконфигурированию** даже при отсутствии DHCP-серверов. По умолчанию каждый хост IPv6 может при подключении к сети автоматически конфигурировать интерфейсы, основываясь на MAC-адресе своего сетевого адаптера и информации, предоставляемой соседними маршрутизаторами.

Типы адресов IPv6

В IPv6 предусмотрены три основных типа адресов (RFC 4291):

- **Индивидуальный адрес** (unicast), как и в версии IPv4, является уникальным идентификатором отдельного интерфейса конечного узла или маршрутизатора. Существуют несколько типов индивидуальных адресов. Некоторые из них (site-local, IPv4-Compatible адреса) уже успели устареть и не рекомендуются к использованию в новых разработках, другие — как, например, глобальный индивидуальный адрес или адрес уровня линии связи (link-local unicast) — активно используются.
- **Групповой адрес** (multicast) аналогичен по назначению групповому адресу IPv4 — он идентифицирует группу интерфейсов, относящихся, как правило, к разным узлам. Пакет с таким адресом доставляется всем интерфейсам, имеющим аналогичный адрес. Групповые адреса в некоторых случаях выполняют функцию отсутствующих в IPv6 широковещательных адресов.
- **Адрес произвольной рассылки** (anycast), как и групповой адрес, определяет группу интерфейсов, но, в отличие от группового адреса, пакет, в поле адреса назначения которого стоит адрес произвольной рассылки, доставляется одному из интерфейсов группы, как правило, ближайшему в соответствии с метрикой, используемой про-

[1] В данном случае термин «линия связи» соответствует определению, данному в разделе «Стек протоколов TCP/ IP» главы 13.

токолами маршрутизации. Синтаксически адрес произвольной рассылки ничем не отличается от индивидуального адреса — он назначается из того же диапазона адресов, что и индивидуальные адреса. Адрес произвольной рассылки может быть назначен только интерфейсам маршрутизатора. Интерфейсы маршрутизаторов, входящие в одну группу адресов произвольной рассылки, имеют помимо индивидуальных адресов еще и общий для всех них адрес произвольной рассылки. Адреса такого типа ориентированы на маршрутизацию от источника, когда маршрут прохождения пакета определяется узлом-отправителем путем указания в поле параметров пакета IP-адресов всех промежуточных маршрутизаторов. Например, провайдер может присвоить всем своим маршрутизаторам один и тот же адрес произвольной рассылки и сообщить его абонентам. Если абонент желает, чтобы его пакеты передавались через сеть этого провайдера, то ему достаточно указать этот адрес произвольной рассылки в цепочке адресов маршрута от источника, и пакет будет передан через ближайший маршрутизатор этого провайдера.

В IPv6 так же, как в IPv4, определены *особые адреса*:

- **Неопределенный адрес** (unspecified address) 0:0:0:0:0:0:0:0 никогда не назначается и обозначает отсутствие адреса. Например, все пакеты, посылаемые хостом, который находится в состоянии инициализации и еще не имеет адреса, содержат в поле адреса источника неопределенный адрес — 128 нулей. Неопределенный адрес не может быть использован в качестве адреса назначения.
- **Адрес обратной петли** (loopback address) 0:0:0:0:0:0:0:1 в IPv6 используется узлом для того, чтобы посылать пакеты самому себе. Пакеты с адресом обратной петли в поле адреса источника или адреса назначения отбрасываются маршрутизаторами.

Тип адреса определяется значением нескольких старших битов адреса, которые называются **префиксом формата (FP)** (табл. 17.1).

Таблица 17.1. Префиксы формата IPv6-адресов

Тип адреса	Двоичная запись префикса формата	Запись префикса в формате IPv6
Неопределенный	00...0 (128бит)	::/128
Адрес обратной петли	00...1 (128 бит)	::1/128
Групповой адрес	11111111	FF00::/8
Адрес уровня линии связи	1111111010	FE80::/10
Глобальный индивидуальный адрес	Все остальные значения	

Индивидуальные адреса

Индивидуальные адреса делятся на:

- **глобальные индивидуальные адреса** (global unicast), предназначенные для использования в интернете; область действия этих адресов не ограничена;
- **индивидуальные адреса уровня линии связи** (link-local unicast), называемые также локальными адресами или адресами уровня подсети, предназначены для локального использования; область действия этих адресов ограничена соответствующей линией связи.

3 бита	45 бит	16 бит	64 бита
001	Глобальный префикс маршрутизации (global routing prefix)	Идентификатор подсети (subnet ID)	Идентификатор интерфейса (interface ID)

Рис. 17.1. Формат глобального индивидуального адреса

Глобальный индивидуальный адрес состоит из трех полей (рис. 17.1).

Глобальный префикс маршрутизации (45 бит) представляет собой общую часть адресов, которыми располагает организация или поставщик услуг интернета, наделяющие адресами клиентов. Глобальный префикс, как правило, имеет иерархическую структуру, состоящую из префиксов адресов провайдеров разного уровня. Эта структура описывает так называемую топологию публичной сети (мы рассматривали этот вопрос при изучении технологии CIDR).

Идентификатор подсети (16 бит) предназначен для адресации подсетей отдельного клиента, например подсетей одной корпоративной сети. Эта работа выполняется администратором сети. Структуризация, выполненная на основе поля идентификатора подсети, отражает топологию частной сети.

Идентификатор интерфейса (64 бита) предназначен для адресации отдельного интерфейса в пределах подсети.

Старшая часть адреса, состоящая из глобального префикса маршрутизации и идентификатора подсети, является аналогом *номера сети* IPv4 и используется для маршрутизации. Оставшиеся 64 бит отводятся под идентификатор интерфейса, который здесь является аналогом *номера узла* в IPv4.

ПРИМЕЧАНИЕ

Можно заметить, что в трех старших битах глобального индивидуального адреса на рис. 17.1 помещен код 001, в то же время из табл. 17.1 следует, что префикс адресов этого типа может принимать любые значения, кроме трех, перечисленных в этой таблице. Это противоречие объясняется тем, что IANA (Internet Assigned Numbers Authority) — центральный орган Интернета, ответственный за распределение адресов, принял решение о резервировании большей части (85 %) адресного пространства IPv6 для будущих применений, оставив для текущего использования «только» блок адресов с первыми битами 001. Поэтому в современном Интернете большая часть глобальных индивидуальных адресов IPv6 имеет префикс 2001::/16.

Отметим, что, в отличие от системы адресации IPv4, в которой размерность поля номера узла является переменной и зависит от того, каким образом сеть клиента разделена на подсети, в IPv6 *размерность поля идентификатора интерфейса фиксирована* и всегда равна 64 битам. В IPv6-адресах в полное распоряжение клиента поступают 2 байта для нумерации сетей и 8 байт для нумерации узлов. Имея такой огромный диапазон адресов, администратор получает широкие возможности. Для сравнительно небольшой сети он может выбрать плоскую организацию, назначая каждой имеющейся подсети произвольные неповторяющиеся значения из диапазона в 65 535 адресов, игнорируя оставшиеся. В крупных сетях более эффективным способом (сокращающим размеры таблиц корпоративных маршрутизаторов) может оказаться иерархическая структуризация сети на основе *агрегирования адресов*. В этом случае используется та же технология CIDR, но уже не поставщиком услуг, а администратором корпоративной сети. Очевидно, что при таком изобилии сетей, которое предоставляется клиенту в IPv6, совершенно теряет смысл операция использования масок

для разделения сетей на подсети, в то время как обратная процедура — объединение подсетей — приобретает особое значение.

Однако чрезмерно большая размерность поля идентификатора интерфейса рассматривается разработчиками технологии IPv6 не только и не столько как возможность нумерации большого числа интерфейсов, но как основа для работы процедур автоматической конфигурации адресов, а также для предоставления администратору большей свободы при назначении адресов.

В то время как администратор IPv4 обычно назначает адреса интерфейсам в виде непрерывной последовательности чисел из диапазона адресов, имеющегося в его распоряжении, администратор IPv6 может позволить себе следовать любому правилу генерирования адресов для интерфейсов при выполнении только двух условий: они должны состоять из 64 битов и не должны повторяться. Для их создания даже может использоваться генератор случайных чисел. Идентификатор интерфейса может быть назначен вручную администратором, либо получен от DHCP-сервера, либо сгенерирован автоматически на основе MAC-адреса сетевого адаптера. Идентификатор интерфейса имеет длину, позволяющую поместить туда MAC-адреса стандартов IEEE 802 (48 бит) или EUI-64 (64 бита), адрес конечного узла ATM (48 бит) или просто серийный номер устройства.

При **автоматическом конфигурировании интерфейсов** младшую часть адреса — идентификатор интерфейса — узел узнает от аппаратуры (сетевого адаптера и т. п.), а старшую — префикс — ему сообщает маршрутизатор. Процедура преобразования MAC-адреса в идентификатор интерфейса (рис. 17.2) является несколько более сложной, чем просто размещение 6 байт MAC-адреса в 8-байтовом поле идентификатора интерфейса.

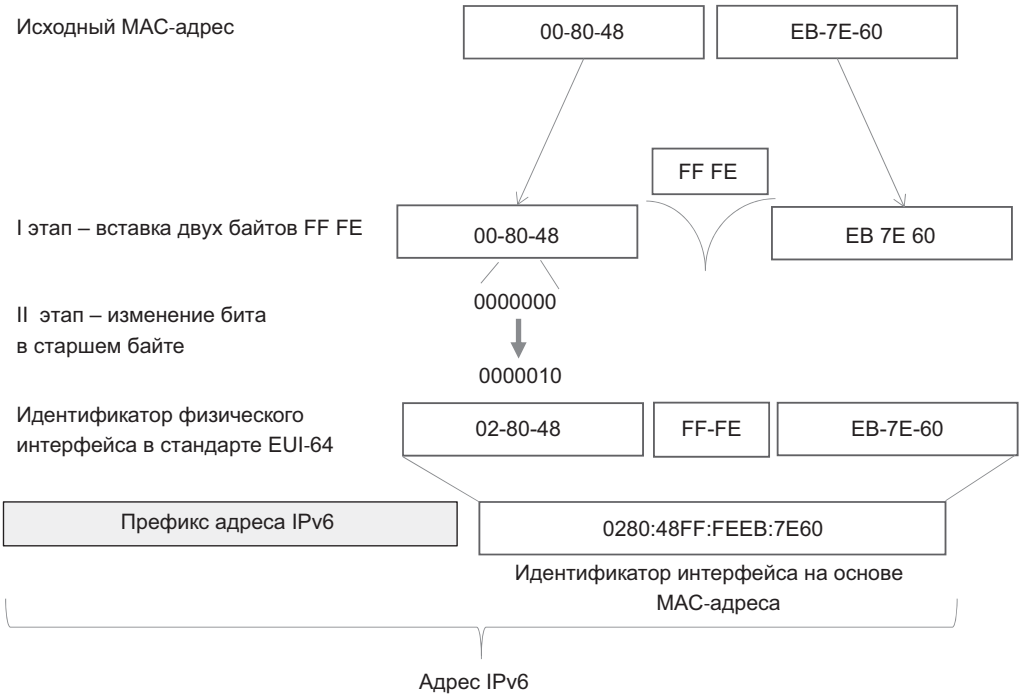


Рис. 17.2. Отображение MAC-адреса на идентификатор интерфейса

Во-первых, между двумя частями MAC-адреса вставляются дополнительные два байта FF и FE, и во-вторых, меняется значение одного бита старшего байта. Такое усложнение объясняется тем, что два десятка лет назад, когда возникли первые опасения относительно возможного истощения 6-байтового адресного пространства MAC-адресов, институт IEEE разработал другой стандарт адресации сетевых интерфейсов — EUI-64 64-Bit (Extended Unique Identifier), в соответствии с которым в качестве канальных адресов используются 8-байтовые числа. Этот стандарт все еще не поддерживается технологией Ethernet — адреса всех сетевых адаптеров и соответствующие поля в кадре Ethernet по-прежнему имеют длину 6 байт, однако он применяется в IPv6 для получения идентификатора интерфейса (RFC 2373).

Итак, после выполнения описанных манипуляций над MAC-адресом 00-80-48-EB-7E-60 получается уникальный идентификатор интерфейса 280:48FF:FEEB:7E60. Добавив к идентификатору интерфейса префикс сети, например 2001:718::/64, система автоконфигурирования получает глобальный уникальный адрес 2001:718::280:48FF:FEEB:7E60.

Другим типом *индивидуальных* IPv6-адресов являются **адреса уровня линии связи** (рис. 17.3). Адреса данного типа не маршрутизируются — они используются локально, то есть при передаче трафика в пределах подсети. Областью действия (scope) адреса уровня линии связи является эта же линия связи. Они автоматически назначаются *каждому* интерфейсу IPv6, даже тому, который не имеет глобального индивидуального адреса[1]. Для рассмотренного выше примера, в котором идентификатор интерфейса имеет значение 0280:48FF:FEEB:7E60, индивидуальный адрес уровня линии связи будет выглядеть следующим образом: **FE80**::0280:48FF:FEEB:7E60, где FE80::/64 — стандартный префикс формата для адресов данного типа.



Рис. 17.3. Формат адреса уровня линии связи

Подобно частным адресам протокола IPv4 (10.0.0.0/8, 172.16.0.0/12 и 192.168.0.0/16), в шестой версии вводятся **уникальные адреса для локального использования** (unique local addresses) с префиксом **FC00::/7**. Областью действия этих адресов является внутренняя сеть предприятия, они не управляются централизованно и не маршрутизируются во внешних сетях.

Групповые адреса

Групповые адреса в IPv6 (RFC 7346) используются таким же образом, что и в IPv4, но играют более важную роль, хотя бы потому, что здесь они используются вместо широковещательной рассылки. На рис. 17.4 показан формат группового адреса.

[1] Адреса уровня линии связи IPv6 аналогичны адресам APIPA (Automatic Private IP Addressing), используемым в ОС Microsoft для автоматического конфигурирования интерфейсов.

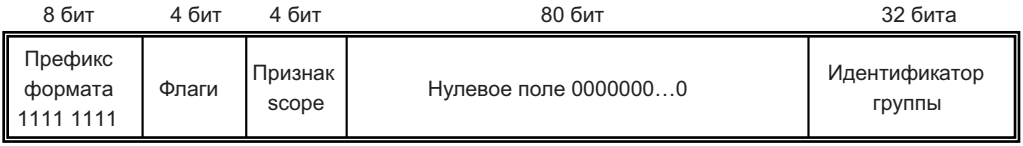


Рис. 17. 4. Формат группового адреса IPv6

Поле *флагов* состоит из 4 бит, из которых в настоящее время используется только один: установленный в 1, он указывает, что номер группы в этом адресе является постоянным, а в 0 — временным.

В IPv6 групповой адрес имеет явно указываемый признак *scope*, отсутствующий в групповом адресе версии IPv4 (табл. 17.2). Значения первых двух признаков, interface-local и link-local, устанавливаются автоматически, а двух оставшихся — вручную, в процессе конфигурирования линий связи.

Таблица. 17.2. Области действия групповых адресов

scope	Область действия адреса
1	interface-local scope — область действия, ограниченная одним интерфейсом
2	link-local scope — область действия, ограниченная линией связи
8	organization-local scope — область, включающая несколько сайтов, относящихся к одной организации
E	global scope — область действия не имеет ограничений, адреса действительны во всем Интернете IPv6

Идентификатор группы может быть постоянным (well known) или временным (transient). Примером постоянной группы является ::1 (все узлы) и ::2 (все маршрутизаторы). С учетом префикса и значения битов, указывающих на область действия группового адреса, получаем:

FF02::1 — адрес группы, состоящей из всех узлов линии связи;

FF02::2 — адрес группы, состоящей из всех маршрутизаторов линии связи.

Из определения следует, что адрес FF02::1 заменяет *широковещательный адрес 255.255.255.255 IPv4*.

Изначально стандарт определял поле идентификатора группы длиной 112 бит. Но затем была выпущена рекомендация, ограничивающая это поле 32 битами. Это связано с тем, что групповая рассылка включает отображение группового адреса сетевого уровня на *групповой MAC-адрес*. Для этих целей в адресном пространстве Ethernet выделен специальный диапазон групповых адресов — MAC-адреса вида **33-33-xx-xx-xx-xx**, где младшие 4 байта отводятся под младшие 4 байта группового адреса IPv6. Таким образом, уменьшение числа возможных номеров групп с 2^{112} до 2^{32} позволило однозначно отображать номера групп на MAC-адреса.

Особым, не существующим в IPv4 видом группового адреса является **групповой адрес запрашиваемого узла** (Solicited-Node Multicast Address, **SNMA**). Адрес этого типа используется в тех случаях, когда некоторый узел делает запрос к другому узлу, для которого он знает только индивидуальный IP-адрес, но не знает MAC-адрес. В IPv4 такого рода запросы посылаются с широковещательным MAC-адресом. В результате все сетевые адаптеры в локальном сегменте получают кадр с этим запросом, извлекают IP-пакет, передают его

«наверх» для анализа IP-адреса назначения. После этого все узлы, кроме узла назначения, отбрасывают пакет. Таким образом, запрос «беспокоит» все узлы данного сегмента локальной сети (включая даже те, на которых не установлен протокол IP).

В IPv6 данная процедура выполняется более эффективно. Вместо широковещания на канальном уровне здесь используется особый вид групповой рассылки. Запрашиваемому узлу (solicited node) назначается групповой IP-адрес SNMA, определяющий группу, в которую с высокой степенью вероятности входит только один этот узел. Как и всякий другой групповой IP-адрес, адрес SNMA отображается на соответствующий групповой MAC-адрес. Несколько упрощая, можно сказать, что запрос, у которого в качестве адресов назначения сетевого и канального уровня указаны эти два групповых адреса, обрабатывается одним сетевым адаптером, что позволяет избежать излишнего вмешательства в работу всех остальных узлов на данной линии связи.

Групповой адрес SNMA автоматически генерируется для каждого индивидуального адреса, назначенного интерфейсу, путем присоединения префикса FF02:0:0:0:0:1:FF00::/104 к трем его младшим байтам. По значению префикса можно увидеть, что адреса этого типа являются локальными, то есть действительны в области, ограниченной одной линией связи. Если некоторому интерфейсу назначается *индивидуальный* адрес 2001:630::0A98:7654:3210, то он сразу получает и соответствующий *групповой адрес запрашиваемого узла*, который в данном случае равен FF02:0:0:0:1:FF54:3210. Тем самым интерфейс автоматически становится членом группы, идентификатор которой включает часть его индивидуального адреса. Эта группа, скорее всего, состоит только из него самого, хотя вероятность того, что в данной локальной сети может обнаружиться другой узел, у которого три младших байта имеют то же самое значение, не нулевая.

Типичный набор адресов интерфейса IPv6

В табл. 17.3 приведен типичный набор адресов, которые в сети IPv6 имеет интерфейс хоста. Помимо компьютера в качестве хоста выступают, например, принтер, телефон или другое устройство, которое не осуществляет маршрутизацию. Примеры адресов даны исходя из следующего: хост оснащен одним сетевым адаптером с MAC-адресом 00:2A:0F:32:5E:D1, состоит в двух подсетях — 2001:A:B:C::/64 и 2001:A:B:1::/64, участвует в группе FF15::1:2:3.

Таблица 17.3. Набор адресов одного интерфейса хоста

Тип адреса	Пример адреса
Индивидуальный адрес уровня линии связи (link-local unicast)	FE80::22A:FFF:FE32:5ED1
Назначенные или сконфигурированные индивидуальные адреса (unicast)	Два сконфигурированных адреса 2001:A:B:C:22A:FFF:FE32:5ED1 и 2001:A:B:1:22A:FFF:FE32:5ED1
Назначенные или сконфигурированные групповые адреса (multicast)	Сконфигурированный групповой адрес FF15::1:2:3
Адрес обратной петли (loopback)	::1
Групповой адрес для стандартной группы «все узлы» (all-nodes multicast)	FF01::1 (область действия — интерфейс) и FF02::1 (область действия — линия связи)
Групповой адрес запрашиваемого узла (solicited-node multicast)	FF02::1:FF32:5ED1

Интерфейсы маршрутизаторов, кроме перечисленных в таблице, имеют дополнительные адреса, в число которых входят групповые адреса «все маршрутизаторы» и адреса произвольной рассылки (anycast). Наличие большого числа адресов для одного интерфейса рождает новые проблемы. Какой из адресов должен быть использован в одном случае, а какой — в другом? Некоторые правила, которыми следует руководствоваться при ответе на этот вопрос, приводятся в RFC 3484. Далее, в разделе «Протокол обнаружения соседей Neighbour Discovery», мы рассмотрим пример использования разных типов адресов.

Формат пакета IPv6

Одной из основных целей изменения формата заголовка протокола IPv6 было сокращение накладных расходов, то есть уменьшение объема служебной информации, передаваемой с каждым пакетом. Для этого в новом протоколе IP были введены понятия *основного* и *дополнительных* заголовков. Основной заголовок присутствует всегда, а необязательные дополнительные заголовки включаются в пакет при необходимости. Они могут содержать, например, информацию о фрагментации пакета, полный маршрут следования пакета при маршрутизации от источника, информацию, необходимую для защиты передаваемых данных, и др.

На рис. 17.5 показан пример структуры пакета IPv6, в котором представлены *все* дополнительные заголовки. В действительности пакеты IPv6 могут иметь один, несколько или

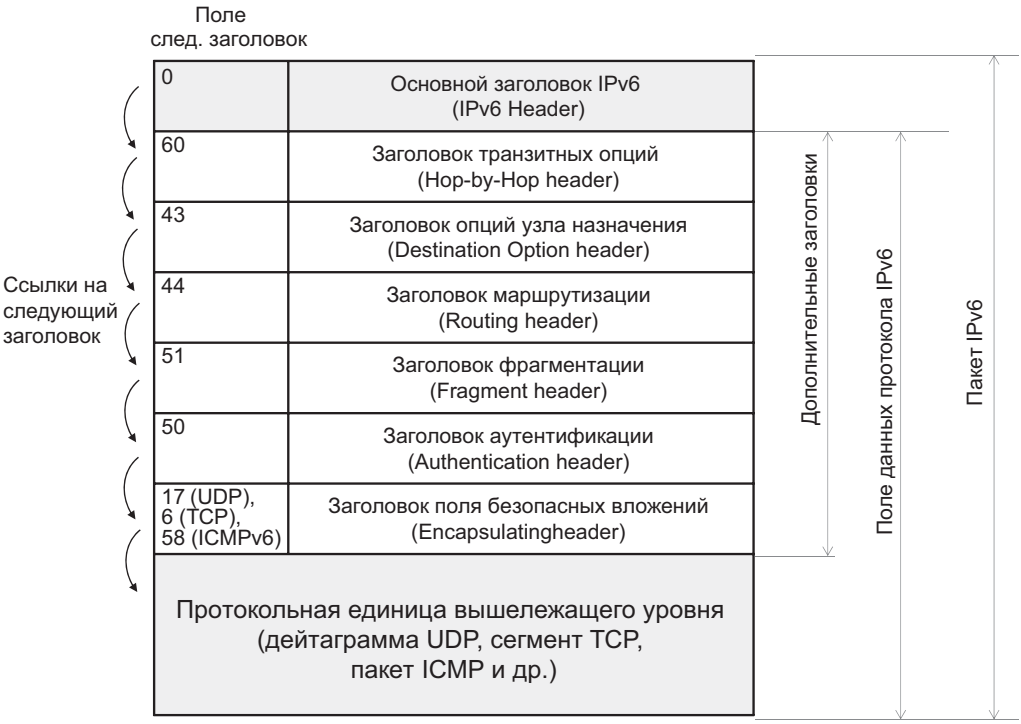


Рис. 17. 5. Структура пакета IPv6

совсем не иметь дополнительных заголовков. Каждый заголовок в этой последовательности указывает на то, какой именно заголовок должен идти следующим, поместив в своем поле *«Следующий заголовок» (next header)* код, закрепленный за данным типом заголовка. Так, код 0 в основном заголовке указывает на то, что за ним должен следовать заголовок транзитных опций, а код 60 в заголовке транзитных опций указывает на заголовок опций узла назначения.

Основной заголовок

Основной заголовок имеет фиксированную длину в 40 байт и включает следующие поля (рис. 17.6):

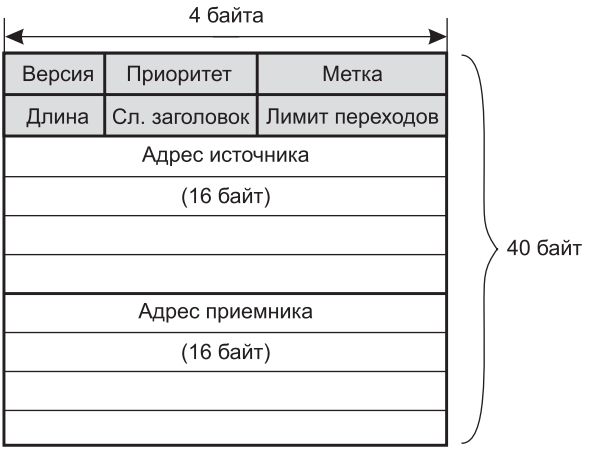


Рис. 17. 6. Формат основного заголовка

- *Версия* протокола (version) занимает 4 бита и равно номеру версии — 6.
- *Класс трафика* (traffic class) занимает 8 бит. Это поле аналогично полю «Тип сервиса» в IPv4 и может быть использовано для разбиения трафика на небольшое количество агрегированных классов обслуживания с различным уровнем QoS.
- *Метка* потока (flow label, 20 бит) указывает на принадлежность пакета к тому или иному потоку. Поле «Метка потока», *отсутствующее* в протоколе IPv4, позволяет выделить из общего трафика индивидуальные потоки и обслуживать их отличным от других пакетов способом. Отличие может заключаться в обеспечении индивидуального уровня QoS или же в индивидуальном маршруте потока. В узле-источнике всем пакетам потока, для которого требуется особое обслуживание, присваивается значение метки, вычисленное как ненулевое псевдослучайное число.

  Маршрутизатор, получив пакет с ненулевым и неизвестным ему значением поля метки, обрабатывает этот пакет так же, как и пакеты с нулевой меткой. Он просматривает все заголовки пакета и на основе полученной информации выполняет требуемые действия. Например, по адресу назначения маршрутизатор определяет адрес следующего маршрутизатора, анализируя поле класса трафика, решает, в какую приоритетную очередь поместить пакет, и т. п. Затем на основании предписанной ему

политики маршрутизатор решает, обслуживать ли все пакеты с такой меткой как поток. Если да, то маршрутизатор делает в кэш-памяти запись, содержащую сведения об обработке этого пакета, позволяющие выполнить обработку следующих пакетов с этой же меткой ускоренным образом. Запись индексируется значением метки и адресом источника. Когда следующий пакет потока поступает на маршрутизатор, из кэша извлекается соответствующая запись и для пакета выполняется ускоренная процедура обработки, при которой, например, вместо поиска адреса следующего маршрутизатора в таблице он просто извлекается из кэша, а вместо анализа поля класса обслуживания пакет направляется в ту же приоритетную очередь, что и первый пакет потока. Время жизни записи в кэш-памяти ограничено несколькими секундами; затем запись удаляется, метка снова становится неизвестной маршрутизатору, и вся процедура повторяется с самого начала.

- *Длина* поля данных (payload length, 16 бит). Число, которое содержится в этом поле, определяет количество байтов, занимаемое всеми дополнительными заголовками и протокольной единицей данных вышележащего уровня. Максимальное значение этого поля равно 65 535. IPv6 допускает существование пакетов с более длинным полем данных, так называемых сверхбольших дейтаграмм, или *джамбограмм* (jumbograms). В этом случае для задания длины используется 32-разрядное поле одной из опций дополнительного заголовка транзитных опций hop-by-hop.
- *Следующий заголовок* (next header, 8 бит). Это поле соответствует по назначению полю «протокол верхнего уровня» в версии IPv4 и содержит код, определяющий тип заголовка, который следует за текущим. Каждый следующий дополнительный заголовок также содержит поле следующего заголовка. Если IPv6-пакет не содержит дополнительных заголовков, то в этом поле будет значение, закрепленное за протоколом TCP, UDP, RIP, OSPF или другим, определенным в стандарте IPv4.
- *Лимит переходов*, или предельное число шагов (hop limit, 8 бит). Это поле за небольшим исключением аналогично полю времени жизни TTL протокола IPv4: при прохождении каждого маршрутизатора из значения этого поля вычитается единица. Когда оно становится равным 0, маршрутизатор отбрасывает пакет и посылает ICMP-сообщение о превышении предельного числа шагов. Максимальное значение поля равно 255.
- Поля *адреса источника* и *адреса назначения* занимают по 128 бит.

Дополнительные заголовки

В стандартах протокола IPv6 фигурируют следующие типы **дополнительных заголовков**:

- *заголовок транзитных опций* (hop-by-hop options) — параметры, используемые при обработке пакетов маршрутизаторами;
- *заголовок опций места назначения* (destination options) — дополнительная информация для промежуточных или финального узлов назначения;
- *заголовок маршрутизации* (routing) содержит в поле данных последовательность IPv6-адресов всех промежуточных узлов, которые должен пройти пакет на пути от источника до места назначения; такой способ задания маршрута называют маршрутизацией от источника;
- *заголовок фрагментации* (fragment) — информация, относящаяся к фрагментации IP-пакета (поле обрабатывается только в конечных узлах);

- *заголовок аутентификации* (authentication) — информация, необходимая для аутентификации конечных узлов и обеспечения целостности содержимого IP-пакетов;
- *заголовок поля безопасных вложений* (encapsulating security payload) — информация, необходимая для обеспечения конфиденциальности передаваемых данных путем шифрования и дешифрирования.

В каждом узле, лежащем на пути перемещения пакета по сети, дополнительные заголовки обрабатываются строго в той последовательности, в которой они представлены в пакете. Хотя стандарты IPv6 не предъявляют жестких требований к порядку следования заголовков, большинство производителей сетевых средств придерживаются в этом вопросе рекомендаций RFC 8200 (именно в этом рекомендуемом порядке дополнительные заголовки перечислены выше). Составляя рекомендации, разработчики учитывали влияние порядка следования заголовков на производительность обработки пакетов. Так, например, заголовок транзитных опций, если он присутствует, должен всегда идти первым после основного заголовка, поскольку информация, которую он несет, может анализироваться и обрабатываться каждым узлом на пути пакета. Дополнительные заголовки могут иметь как переменную длину (заголовки транзитных опций и опций места назначения), так и фиксированную длину (все остальные заголовки).

Как заголовок транзитных опций, так и заголовок опций места назначения может включать переменное число информационных структур — **опций**, обрабатываемых в порядке их следования в заголовке. Каждая опция состоит из трех полей:

- *Тип опции* состоит из 8-битового кода, который, помимо указания на собственно тип опции, говорит маршрутизатору, может ли в результате обработки этой опции измениться маршрут следования данного пакета, а также что он должен сделать с пакетом, если не распознает тип опции: пропустить опцию, отбросить пакет или отбросить пакет и послать ICMP-сообщение.
- *Длина* поля данных *опции* (8 бит), которая содержит число, указывающее длину опции в байтах без учета длины первых двух полей.
- *Данные опции* — поле переменной длины, содержащее информацию, специфичную для конкретного типа опции.

На рис. 17.7 показан формат заголовка транзитных опций:

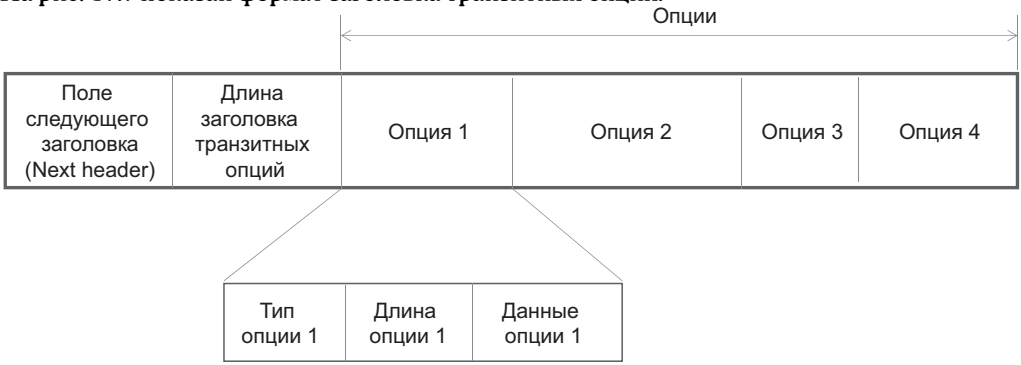


Рис. 17.7. Формат заголовка транзитных опций

Рассмотрим одну из опций этого заголовка — **опцию сверхбольшого поля данных** (Jumbo Payload Option), которая позволяет передавать джамбограммы (IPv6-пакеты,

поле данных которых превышает 65 535 байт, максимально допустимую длину, определяемую основным заголовком). Необходимость в таких пакетах может возникать, например, при передаче данных, производимых суперкомпьютерами. Узел может передавать джамбограммы только при условии, что он подсоединен к сети линией связи, имеющей MTU больше 65 575 (65 535 байт поля данных плюс 40 байт заголовка). Поле длины в основном заголовке пакета, несущем данную опцию, должно быть нулевым. Когда узел IPv6 получает пакет с нулевыми значением длины и нулевым значением в поле «Next header», он понимает, что длину поля данных этого пакета надо определять на основе данных заголовка транзитных опций, который следует сразу за основным заголовком. Опция сверхбольшого поля данных имеет следующие значения полей: тип опции — 194, длина опции — 4 байта, данные опции — число из интервала ($2^{16} - 2^{32}$), равное длине джамбограммы в байтах.

Снижение нагрузки на маршрутизаторы

Поскольку для маршрутизации пакета обязательным является лишь основной заголовок (большинство дополнительных заголовков обрабатывается только в конечных узлах), *это снижает нагрузку на маршрутизаторы*. В то же время возможность использования большого количества дополнительных параметров *расширяет функциональность протокола IP* и делает его открытым для внедрения новых механизмов.

Чтобы повысить производительность маршрутизаторов интернета в части выполнения их основной функции — продвижения пакетов, в версии IPv6 предпринят ряд мер по освобождению маршрутизаторов от некоторых вспомогательных функций.

- *Отказ от обработки необязательных параметров заголовка.* В IPv4 обработка пакета включает просмотр и анализ всех полей заголовка, даже если они не несут полезной информации. Например, при обработке нефрагментированных пакетов просматриваются все поля, относящиеся к фрагментации.
- *Перенесение функций фрагментации с маршрутизаторов на конечные узлы.* Конечные узлы в версии IPv6 обязаны найти минимальное значение MTU вдоль всего пути, соединяющего исходный узел с узлом назначения (эта техника под названием Path MTU Discovery уже используется в IPv4). Маршрутизаторы IPv6 не выполняют фрагментацию, а только посылают ICMP-сообщение о слишком длинном пакете конечному узлу, который должен уменьшить размер пакета.
- *Широкое использование маршрутизации от источника.* При маршрутизации от источника узел-источник задает полный маршрут прохождения пакета через сети. Такая техника освобождает маршрутизаторы от необходимости просмотра адресных таблиц при выборе следующего маршрутизатора.
- *Отказ от подсчета контрольной суммы.* В заголовке пакета IPv6 нет поля контрольной суммы. Поскольку контрольная сумма вычисляется как на вышележащем (TCP, UDP), так и на нижележащем уровне (Ethernet), было решено, что вычисление контрольной суммы на сетевом уровне избыточно.
- *Агрегирование адресов* ведет к уменьшению размера адресных таблиц маршрутизаторов, а значит, и к сокращению времени просмотра и обновления таблиц. При этом также сокращается служебный трафик, порождаемый протоколами маршрутизации.

- *Отказ от использования технологии NAT*[1]. Одной из целей технологии NAT является уменьшение необходимого адресного пространства путем отображения большого числа частных адресов на несколько публичных. В IPv6 такая экономия адресов теряет смысл. Отказ от NAT упрощает маршрутизацию, дает выигрыш в производительности, решает проблему идентификации пользователей.
- *Принципиальная возможность использования в качестве номера узла его MAC-адреса* избавляет маршрутизаторы от необходимости применять процедуру разрешения адресов.

Новая версия протокола IP, являющаяся составной частью проекта IPv6, предлагает встроенные средства защиты данных. Размещение средств защиты на сетевом уровне делает их прозрачными для приложений, так как между уровнем IP и приложением всегда будет работать протокол транспортного уровня. Приложения переписывать при этом не придется. Новая версия протокола IP со встроенными средствами обеспечения безопасности называется **IPSec** (Security Internet Protocol — защищенный протокол IP, подробнее см. главу 31).

Протокол обнаружения соседей Neighbour Discovery

Протокол IPv6 не может работать без вспомогательных протоколов, выполняющих функции отображения адресов и имен, оповещения о произошедших в сети ошибках, снабжающих приложения диагностической информацией о работе сети, а также протоколов, поддерживающих конфигурирование сетевых узлов. В IPv4 такую роль играют протоколы ARP, DNS, ICMP, DHCP и др. Для работы с IPv6 одни из этих протоколов были подвергнуты минимальной модернизации, другие — заменены новыми средствами. Таким новым протоколом, пришедшим на смену протоколу ARP и заменившим функции обнаружения маршрутизатора (Router Discovery) и перенаправления (Redirect) протокола ICMP, является **протокол обнаружения соседей** (Neighbor Discovery, **ND**).

Задачи протокола ND и протокол ICMPv6

Протокол обнаружения соседей (Neighbor Discovery, **ND**) описан в RFC 2461. Он представляет собой набор сообщений и процессов, определяющих взаимоотношения между смежными узлами в ходе решения следующих задач:

- ❑ разрешение адресов — аналог функций протокола ARP;
- ❑ обнаружение хостом соседних маршрутизаторов, выбор маршрутизатора по умолчанию — аналог функции обнаружения маршрутизатора ICMP IPV4 Router Discovery;
- ❑ наделение хостов параметрами MTU линии связи, значение предельного числа шагов по умолчанию для отправляемых хостом пакетов, значение префикса для адресов уровня линии связи;
- ❑ автоконфигурирование IP-адресов;
- ❑ обнаружение дублированных адресов;
- ❑ перенаправление маршрута (аналог функции ICMP Redirect в IPv4).

[1] О технологии NAT читайте в главе 30.

Для решения перечисленных задач узлы, относящиеся к одной линии связи, обмениваются специально предназначенными для этих целей сообщениями ICMPv6, из чего следует, что основой для протокола ND служит **протокол ICMPv6** (RFC 4443), входящий в состав программного обеспечения любого узла IPv6. Являясь усовершенствованной версией протокола ICMP для IPv4, протокол ICMPv6 играет в IPv6 значительно бóльшую роль, чем ICMP в IPv4. Действительно, помимо работы по оповещению об ошибках и выработке диагностических сообщений, ICMPv6 предоставляет ряд своих сообщений для реализации протокола обнаружения соседей ND и некоторых других протоколов стека IPv6.

Главной идеей модернизации протокола ICMP было стремление избавиться от редко используемых и избыточных механизмов и одновременно расширить функциональность протокола для поддержки новых возможностей IPv6. Многое в новой версии сохранилось без изменения: формат сообщений (см. вновь рис. 14.14); разделение сообщений на две группы — сообщения об ошибках и информационные эхо-сообщения; названия и назначение многих сообщений. В то же время имеются и изменения, как менее, так и более значительные. Так, изменена нумерация типов сообщений — даже тех, названия которых полностью совпадают со старыми; некоторые сообщения, сохранив свое название, изменили, хотя и незначительно, интерпретацию. Так, сообщение «Истечение времени» в IPv4 интерпретировалось как превышение времени жизни пакета, а в IPv6 — как превышение числа хопов. Введены новые типы и коды сообщений, отражающие специфику IPv6, например сообщение «Слишком большой пакет», отсутствующее в ICMP для IPv4. В сообщении о недоступности адресата появилась новая причина ошибки: «Адрес назначения пакета выходит за рамки области действия адреса источника»; это сообщение генерируется, когда, например, в качестве адреса источника используется адрес уровня линии связи, а в качестве адреса назначения — глобальный индивидуальный адрес. Для сообщения типа «Проблема с параметрами пакета» определен новый код ошибки («Ошибка в распознавании опций пакета IPv6»).

Сообщения протокола ND

Основными «кирпичиками», из которых построен протокол обнаружения соседей ND, являются следующие пять типов сообщений ICMP, называемых также сообщениями ND:

1. Сообщение **«Запрос к маршрутизатору»** (Router Solicitation), код типа 133, посылается хостом по групповому адресу FF02::2, когда он желает узнать, имеются ли маршрутизаторы, непосредственно подсоединенные к его линии связи. Хост посылает это сообщение, приглашая все маршрутизаторы ответить *немедленно*, не ожидая истечения установленных для них интервалов между объявлениями.
2. Сообщение **«Объявление маршрутизатора»** (Router Advertisement), код типа 134, может рассылаться маршрутизатором либо периодически с некоторым интервалом по групповому адресу FF02::1 (всем узлам на его линии связи), либо в ответ на поступивший «запрос к маршрутизатору» по адресу запросившего информацию узла. Это сообщение содержит достаточно обширный перечень необходимых хосту данных, в частности, информацию о значении одного или нескольких префиксов для индивидуальных адресов, о значении MTU данной линии связи, информацию о специфических маршрутах, а также о том, следует ли узлу-получателю данного сообщения использовать автоконфигурацию, и если да, то как долго будут действительны адреса, сгенерированные при автоконфигурации. Для того чтобы узел мог определить, какой

из маршрутизаторов в его подсети следует использовать в качестве *маршрутизатора по умолчанию*, в объявлении маршрутизатора предусмотрен двухбитовый параметр, описывающий степень предпочтительности данного маршрутизатора в роли маршрутизатора по умолчанию: 00 — средняя; 01 — высокая; 11 — низкая; 00 — маршрутизатор вообще не может быть использован в этом качестве. Эти рейтинги назначаются маршрутизаторам администраторами при конфигурировании с учетом особенностей конкретной сети.

3. Хост посылает сообщение **«Запрос к соседнему узлу»** (Neighbor Solicitation), код типа 135, чтобы обнаружить наличие дубликата при назначении адреса, узнать адрес канального уровня соседнего узла или проверить достижимость соседнего узла.
4. Сообщение **«Объявление соседнего узла»** (Neighbor Advertisement), код типа 136, посылается узлом либо в ответ на сообщение «Запрос к соседнему узлу» по индивидуальному адресу запрашивающего узла, либо по собственной инициативе данного узла в тех случаях, когда изменился его MAC-адрес или изменилась роль, которую он играет в сети (маршрутизатор или хост). В последнем случае сообщение посылается широковещательно (по адресу FF02::1 в пределах линии связи).
5. Назначение сообщения **«Перенаправление маршрута»** (Redirect), код типа 137, совпадает с назначением аналогичного сообщения протокола ICMP для IPv4 (см. раздел «Формат, типы и коды ICMP-сообщений» в главе 14).

Все сообщения, которыми оперирует протокол ND, должны поступать *только от узлов-соседей*, то есть узлов, относящихся к одной линии связи. Для того чтобы можно было убедиться, что пакет, несущий сообщение ND, пришел из той же локальной сети, все сообщения протокола ND отправляются с предустановленным значением 255 в поле «Предельное число шагов». Когда сообщение ND поступает в пункт назначения, выполняется проверка: если число шагов в заголовке пакета *меньше 255*, значит, этот пакет прошел через маршрутизатор, то есть пришел извне и должен быть отброшен. В результате протокол ND защищен от ND-сообщений, ошибочно или злонамеренно направленных из внешней сети. В сети IPv4 ICMP-сообщения «Объявление маршрутизатора» и «Перенаправление маршрута» могут поступать в локальный сегмент извне, что также представляет собой потенциальную угрозу.

Каждое из сообщений ND может иметь параметры, называемые *опциями протокола ND*. Каждый тип опции несет в себе определенный вид параметра: префикс сети или MAC-адрес, MTU для линии связи или временной интервал между объявлениями маршрутизатора и др. Один и тот же тип опции может использоваться в сообщениях разного типа, например, опция «Адрес канального уровня источника» может быть включена как в запрос к маршрутизатору, так и в объявление соседнего узла.

На рис. 17.8, *а* показан формат сообщения протокола ND, в поле данных которого размещается набор опций. Сообщение может иметь одну или несколько опций либо не иметь опций вообще. На рис. 17.8, *б* показан пример IPv6-пакета с инкапсулированным в него ICMPv6-сообщением. Опции в сообщениях ND имеют такой же формат, как и опции в дополнительных заголовках (рис. 17.7), то есть состоят из трех полей: «Тип опции», «Длина опции» и «Данные опции» с той разницей, что здесь длина опции задается не в байтах, а в 8-байтных блоках. В примере число 135 в поле типа сообщения указывает на то, что это сообщение «Запрос к соседнему узлу». В поле «Тип опции» стоит 1, что указывает на опцию «Адрес канального уровня отправителя», длина которой составляет один 8-байтовый блок, в поле данных этой опции размещается MAC-адрес отправителя.

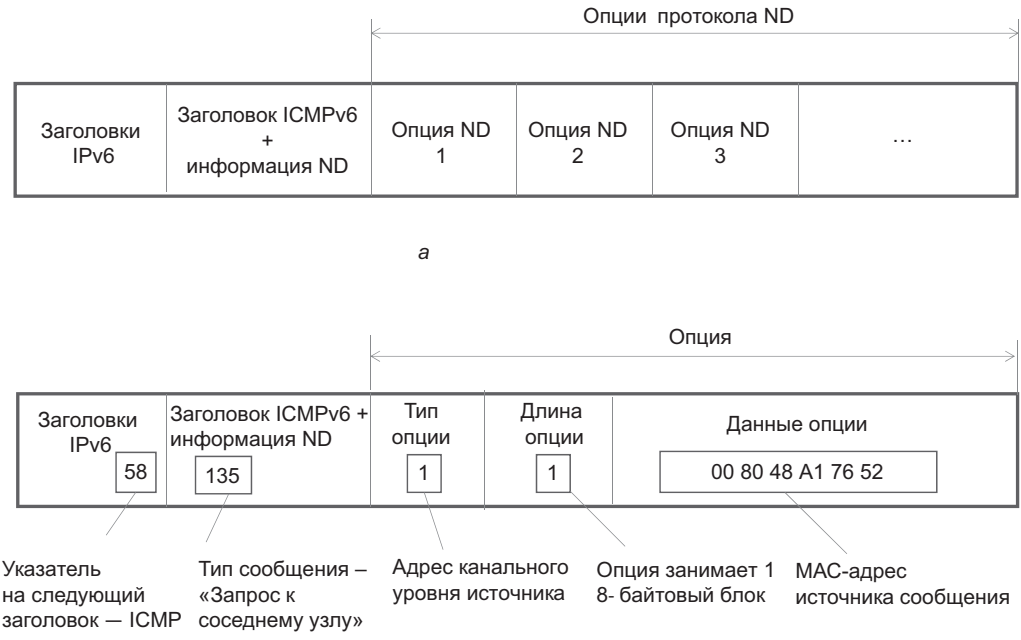


Рис. 17.8. Формат сообщения ND: *а* — обобщенный формат; *б* — формат сообщения «Запрос к соседнему узлу»

Проверка наличия дубликата адреса с помощью протокола ND

При конфигурировании интерфейса узел должен убедиться, что в его подсети (линии связи) нет другого узла с таким же адресом. Для этого он посылает всем своим соседям сообщение типа «Запрос к соседнему узлу», содержащее значение IP-адреса, который проверяется на наличие дубликата. Если в ответ не приходит сообщения «Объявление соседнего узла», то адрес может быть использован. Рассмотрим пример решения задачи поиска дубликата адреса (рис. 17.9). Пусть интерфейс узла А имеет MAC-адрес 00-80-48-EB-7E-60 и *неподтвержденный* индивидуальный адрес 2001:630:3C:F803::7. Для проверки на наличие дубликата узел А посылает сообщение «Запрос к соседнему узлу» со следующими значениями полей:

Заголовок сообщения «Запрос соседнего узла» — целевой адрес 2001:630:3C:F803::7.

Заголовок IP-пакета:

- IP-адрес отправителя — неопределенный адрес (::), так как неподтвержденный адрес нельзя использовать ни для отправки, ни для получения пакетов.
- IP-адрес назначения — групповой адрес SNMA FF02::1:FF00:7, который получается из целевого адреса 2001:630:3C:F803::7. Как мы знаем, адреса такого типа используются, когда MAC-адрес узла назначения неизвестен, что мы и имеем в данном случае.

Заголовок кадра Ethernet:

- MAC-адрес отправителя — 00-80-48-EB-7E-60 — MAC-адрес узла А;
- MAC-адрес получателя — 33-33-FF-00-00-07, групповой MAC-адрес, полученный отображением группового адреса SNMA запрошенного узла FF02::1:FF00:7.

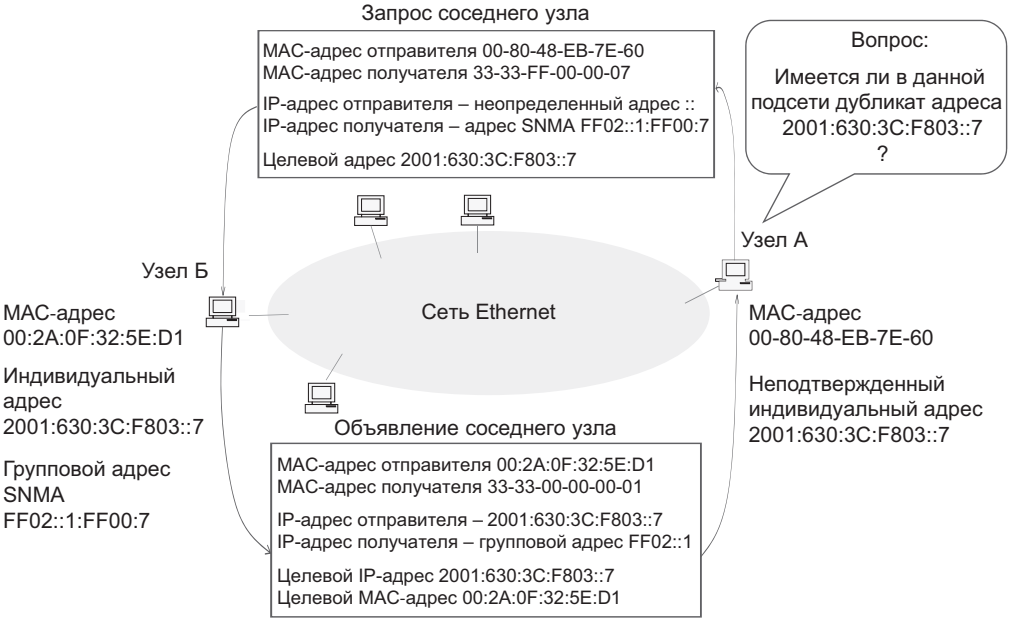


Рис. 17.9. Обмен сообщениями ND для проверки наличия дубликата адреса

Данное сообщение принимается и обрабатывается интерфейсом узла В, так как в число его адресов входят групповые адреса SNMA FF02::1:FF00:7 и 33-33-FF-00-00-07. В соответствии с протоколом Neighbor Discovery узел В отвечает сообщением «Объявление соседнего узла» со следующими значениями полей:

Заголовок сообщения «Объявление соседнего узла» — целевой адрес 2001:630:3C:F803::7.

Опция протокола Neighbor Discovery — целевой адрес канального уровня — 00:2A:0F:32:5E:D1.

Заголовок IP-пакета:

- IP-адрес отправителя — 2001:630:3C:F803::7, IP-адрес узла В;
- IP-адрес назначения — групповой адрес FF02::1, указывающий на все узлы в данной линии связи, так как в поле адреса отправителя запроса был указан неопределенный адрес.

Заголовок кадра Ethernet:

- MAC-адрес отправителя — 00:2A:0F:32:5E:D1 — MAC-адрес узла А;
- MAC-адрес получателя — 33-33-00-00-00-01, групповой MAC-адрес «все узлы» подсети, поскольку IP-адрес получателя неизвестен.

В результате данного обмена сообщениями узел А убедился, что его неподтвержденный адрес имеет дубликат, а значит, не может быть использован.

Разрешение адресов в IPv6

Когда по IP-адресу назначения пакета узел-отправитель определяет IP-адрес следующего маршрутизатора, этого недостаточно, чтобы отправить пакет по маршруту, так как пакет требуется поместить в кадр Ethernet, а для этого, помимо IP-адреса следующего маршрутизатора, необходимо знать и его MAC-адрес. В IPv4 функция определения канального адреса интерфейса по его IP-адресу возложена на протокол ARP, а в IPv6 эта задача, как и задача обнаружения дубликата адреса, решается путем обмена сообщениями «Запрос соседнего узла» и «Объявление соседнего узла» (рис. 17.10).

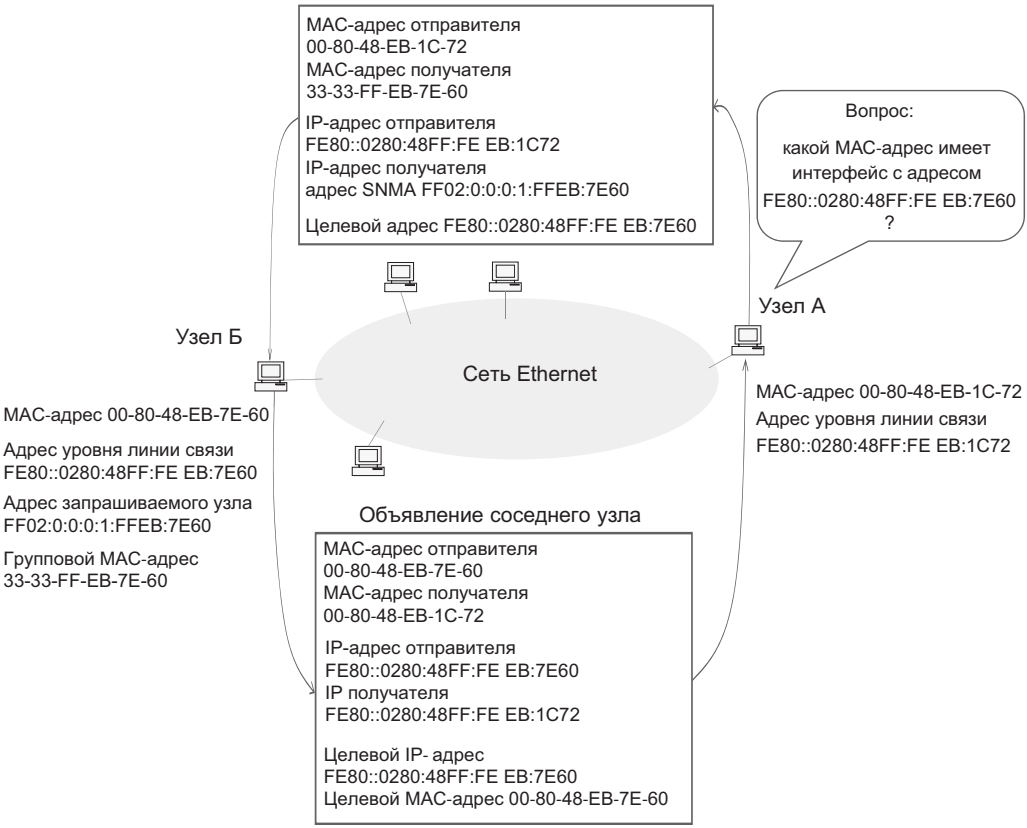


Рис. 17.10. Обмен сообщениями ND для разрешения адреса

Пусть узел *А* решает послать пакет своему соседу по локальной сети — узлу *Б*. Узлу *А* известен индивидуальный IP-адрес узла *В* — FE80::0280:48FF:FE EB:7E60, но он не знает его MAC-адреса. Для разрешения ситуации он посылает в свою локальную сеть сообщение протокола ND «Запрос соседнего узла».

Поскольку узел *А* не может направить это сообщение по индивидуальному адресу узла *Б*, то он решает воспользоваться групповым адресом запрашиваемого узла (SNMA), который конструирует из известного ему индивидуального IP-адреса FE80::0280:48FF:FEEB:7E60

узла *Б*. Полученный в результате адрес FF02:0:0:0:1:FFEB:7E60 описывает группу, скорее всего, состоящую из одного узла *Б*. В качестве MAC-адреса назначения узел *А* использует отображение группового IP-адреса FF02:0:0:0:1:FFEB:7E60 на групповые MAC-адреса — 33-33-FF-EB-7E-60.

Узел *Б* получает «Запрос», так как групповой адрес запрашиваемого узла входит в число его адресов; он был сгенерирован для него одновременно с присвоением индивидуального адреса. Обработав запрос, узел *Б* отправляет ответное сообщение «Объявление соседнего узла» на индивидуальный адрес узла *Б*. В «Объявлении» в поле опции «Адрес канального уровня целевого хоста» узел *А* находит значение искомого MAC-адреса узла *Б*, после чего узлы *А* и *Б* могут связываться по индивидуальным адресам.

Процесс адаптации версии IPv6

Темпы миграции

При разработке IPv6 предполагалось, что довольно значительное время островки интернета, работающие по протоколу IPv6, будут сосуществовать с остальной частью интернета, работающей по протоколу IPv4. Насколько быстро идет процесс разрастания «островков», можно судить по графикам на рис. 17.11 и 17.12.

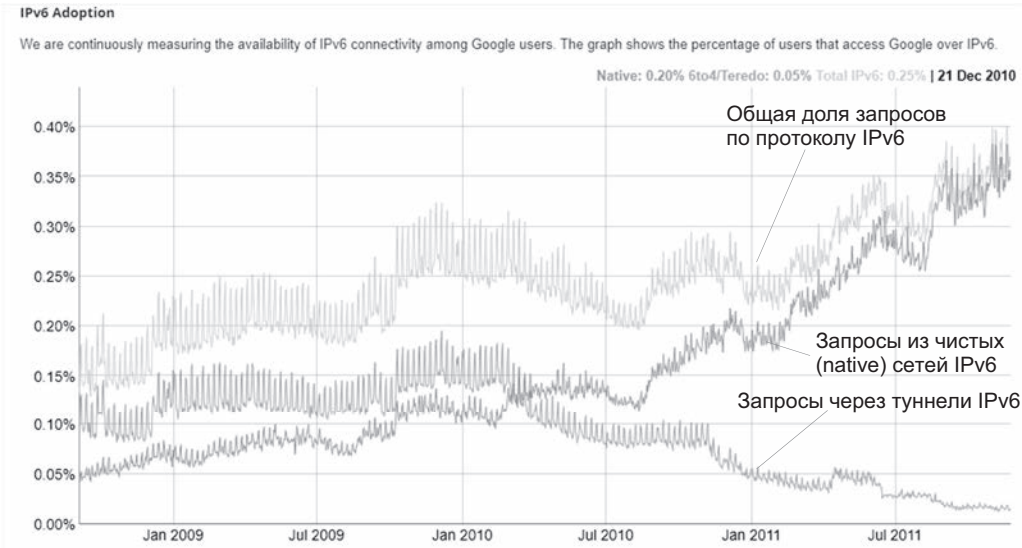


Рис. 17.11. Статистические данные анализа трафика запросов, собранные компанией Google, 2009–2011 год

На рис. 17.11 — графики, показывающие, как изменялась доля запросов к Google, выполняемых по протоколу IPv6, по отношению к общему числу запросов[1]. Один из графиков

[1] https://www.google.com/intl/en/ipv6/statistics.html

описывает запросы через туннели IPv6, другой — запросы из сетей, в которых вся инфраструктура поддерживает IPv6 (так называемых чистых сетей IPv6), третий график представляет собой сумму первых двух. Как видим, в конце 2010 года общая доля IPv6-запросов составляла лишь 0,25 %, причем большая часть (0,2 %) из них поступала из чистых IPv6-сетей, 0,5 % запросов использовали туннелирование, а доля запросов, представляющих все другие методы интеграции, была пренебрежимо малой.

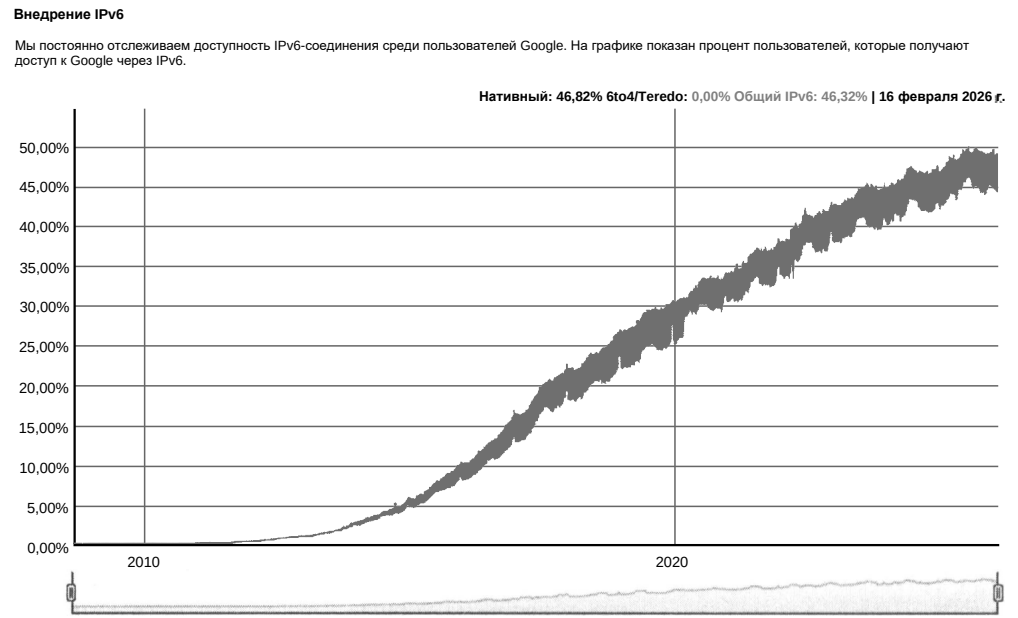


Рис. 17.12. Статистические данные анализа трафика запросов, собранные компанией Google, 2009-2025 годы

Рисунок 17.12 показывает, как сильно изменилась картина через 14 лет — общая доля IPv6-запросов к концу 2025 года достигла почти 50 % (по данным из того же источника)! При этом все запросы приходили из чистых сетей IPv6, туннелирование престало применяться. И хотя эти данные носят частный характер, отражая трафик запросов к поисковой системе Google, на их основе эксперты делают оценки темпов перехода на IPv6. Согласно данным Google, в число стран с набольшим распространением протокола IPv6 входят Франция (86%), Германия (77%), Бельгия (73%), Индия (74%), Саудовская Аравия (70%), Малайзия (65%), США (53%). В РФ это значение составляет около 15 %.

Проблема интеграции сетей разных технологий

Совместное существование сетей, построенных на основе разных технологий, — проблема более общая, нежели сосуществование IPv4 и IPv6. Два десятка лет назад в сетях наблюдалось гораздо большее разнообразие и вопросы согласования технологий в гетерогенных сетях занимали более важное место, чем сейчас. В последнее десятилетие степень неоднородности сетей существенно снизилась. Прежде всего этому способствовал стремительный рост по-

пулярности интернета, в результате чего стандартом де-факто стал стек протоколов TCP/IP. Большое влияние оказали также доминирование на канальном уровне технологии Ethernet и сокращение перечня популярных сетевых ОС до двух семейств — Microsoft Windows и Unix. Таким образом, тенденция к унификации сетевого мира налицо, однако означает ли это, что проблема межсетевого взаимодействия почти решена и на нее не стоит обращать внимания? Правильный ответ — нет. Даже в условиях доминирования стека TCP/IP сегодня существуют две его версии — IPv4 и IPv6, которые не могут непосредственно взаимодействовать друг с другом. Кроме того, учитывая изменчивость сетевого мира, в котором время от времени возникает проблема сосуществования сетей, построенных на разных технологиях, знакомство с основными идеями этих методов может оказаться весьма полезным.

Одной из причин неоднородности любой большой сети является эволюционный характер ее развития. Сеть, как правило, не строится с нуля и не возникает в одно мгновение. За время ее существования появляются привлекательные технологические новшества, и создатели сети вынуждены вносить в нее изменения, которые часто требуют согласования новых технологий с уже имеющимися старыми.

> В контексте межсетевого взаимодействия мы будем понимать под термином «сеть» совокупность компьютеров, общающихся друг с другом с помощью *одного, общего для всех них стека протоколов*.

Проблема возникает, если требуется организовать взаимодействие компьютеров, на которых поддерживаются отличающиеся стеки коммуникационных протоколов. В худшем случае не совпадают протоколы всех уровней, однако проблема межсетевого взаимодействия возникает и тогда, когда в сетях не совпадает протокол хотя бы одного из уровней стека. Так, если в одном стеке на канальном уровне работает технология Ethernrt, а в другом — протокол PPP, то непосредственно эти протоколы общаться и передавать между сетями кадры не могут. Именно для решения подобных проблем и были в свое время созданы протоколы сетевого уровня — IPX, IP и др. Но идея использования протокола сетевого уровня перестает работать, когда в сети на этом уровне работают разные протоколы, например IPv4 и IPv6.

Равным образом проблема межсетевого взаимодействия может возникнуть и при объединении сетей, в которых используется один и тот же протокол сетевого уровня, но применяются различные протоколы прикладного уровня. Например, компьютеры, работающие под управлением ОС Microsoft Windows, по умолчанию используют для доступа к файлам протокол SMB, а компьютеры, работающие под управлением ОС Unix, — протокол NFS. Если же требуется обеспечить пользователям сервис единой разделяемой файловой системы, то администратор сети столкнется с необходимостью согласования сетевых служб.

Двойной стек, трансляция, туннелирование

Исходя из самых общих соображений, можно назвать три принципиально различных подхода к организации взаимодействия узлов, принадлежащих сетям, построенным на основе разных технологий:

- двойной стек протоколов (или мультиплексирование стеков);
- трансляция;
- туннелирование (инкапсуляция).

Для пояснения первых двух подходов может быть использована следующая простая аналогия. Для компьютерных узлов стек коммуникационных протоколов является тем же, что и язык общения для людей. Если вы находитесь в стране, в которой люди говорят на незнакомом для вас языке, то у вас имеется две возможности пообщаться с аборигенами: во-первых, выучить этот язык (установить на компьютере второй стек коммуникационных протоколов), а во-вторых, воспользоваться услугами переводчика (передавать трафик через транслирующее устройство).

Двойной стек. На рис. 17.13 показаны две сети, одна из которых преимущественно использует стек *A* (затемнен фоном), а другая — стек *B*. В сети *A* работает сервер, на котором установлены два стека, *A* и *B*, для того чтобы к нему могли обращаться не только клиенты из его «родной» сети *A*, но и клиенты, использующие стек *B*.

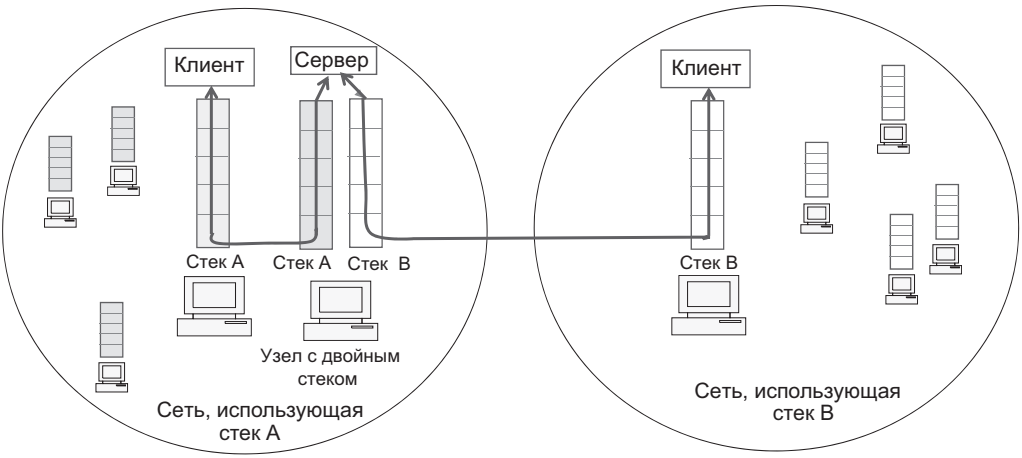


Рис. 17.13. Двойной стек протоколов

Чтобы запрос от прикладного процесса был правильно обработан и направлен через соответствующий стек, необходимо наличие специального программного элемента — **мультиплексора протоколов**, называемого также **менеджером протоколов**. Менеджер должен уметь определять, к какой сети направляется запрос клиента. Для этого может использоваться служба имен сети, в которой отмечается принадлежность того или иного ресурса определенной сети с соответствующим стеком протоколов. В общем случае на каждом уровне вместо одного протокола появляется целый набор протоколов, и может существовать несколько мультиплексоров, выполняющих коммутацию между протоколами разных уровней (рис. 17.14). Хотя многие из показанных на этом рисунке протоколов представляют сейчас только исторический интерес, схема их взаимодействия прекрасно иллюстрирует идею многоуровневого мультиплексирования стеков протоколов.

Мультиплексирование протоколов реализует отношение «один ко многим», то есть один клиент с дополнительным стеком может обращаться ко всем серверам, поддерживающим этот стек; по аналогии один сервер с дополнительным стеком может предоставлять услуги многим клиентам.

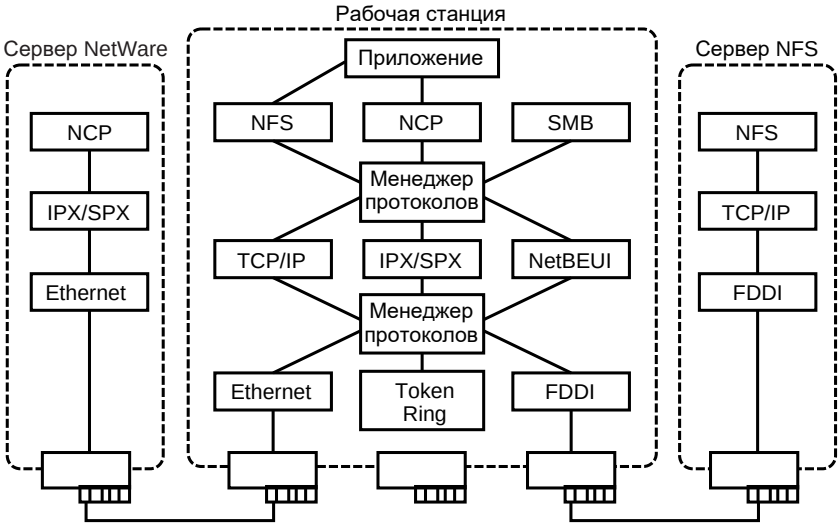


Рис. 17.14. Мультиплексирование протоколов

Трансляция обеспечивает согласование стеков протоколов путем преобразования сообщений, поступающих от одной сети, в формат сообщений другой сети. Транслирующий элемент, в качестве которого могут выступать, например, программный или аппаратный шлюз, мост, коммутатор или маршрутизатор, размещается между взаимодействующими сетями и служит посредником в их «диалоге». В зависимости от типа транслируемых протоколов процедура трансляции может иметь разную степень сложности. Так, преобразование протокола Ethernet в протокол Token Ring сводится к нескольким несложным действиям, главным образом благодаря тому, что оба протокола ориентированы на единую схему адресации узлов. А вот трансляция протоколов сетевого уровня (например, IPv4 в IPv6) представляет собой куда более сложный интеллектуальный процесс, включающий не только преобразование форматов сообщений, но и отображение адресов сетей и узлов, различным образом трактуемых в этих протоколах. Еще более сложной является трансляция протоколов прикладного уровня, включающая отображение инструкций одного протокола на инструкции другого, что представляет собой сложную, логически неоднозначную интеллектуальную процедуру, сравнимую с работой переводчика с одного языка на другой.

На рис. 17.15 показано взаимодействие клиента и сервера, принадлежащих разным сетям, соединенным шлюзом, преобразующим трафик сети *A* в трафик сети *B*. В шлюзе установлены оба стека протоколов. Шлюз транслирует пакеты, поступающие от клиента в адрес сервера, в соответствии с алгоритмом, зависящим от конкретных протоколов и, как отмечено, может быть достаточно сложным.

Преобразованные пакеты передаются в сеть *B* серверу. Ответ сервера клиенту преобразуется шлюзом аналогично. Достоинство шлюзов состоит в том, что они сохраняют в неизменном виде программное обеспечение узлов в сетях как одной, так и другой технологии. Пользователи работают в привычной среде и могут даже не заметить, что получают доступ к ресурсам другой сети. Но, как и всякий централизованный ресурс, шлюз снижает надеж-

ность сети. Кроме того, при обработке запросов в шлюзе возможны относительно большие временны́е задержки, во-первых, из-за затрат времени на собственно процедуру трансляции, во-вторых, из-за задержек запросов в очереди к разделяемому всеми клиентами шлюзу, особенно если запросы поступают с большой интенсивностью. Это делает шлюз плохо масштабируемым решением (хотя ничто не мешает установить в сети несколько параллельно работающих шлюзов).

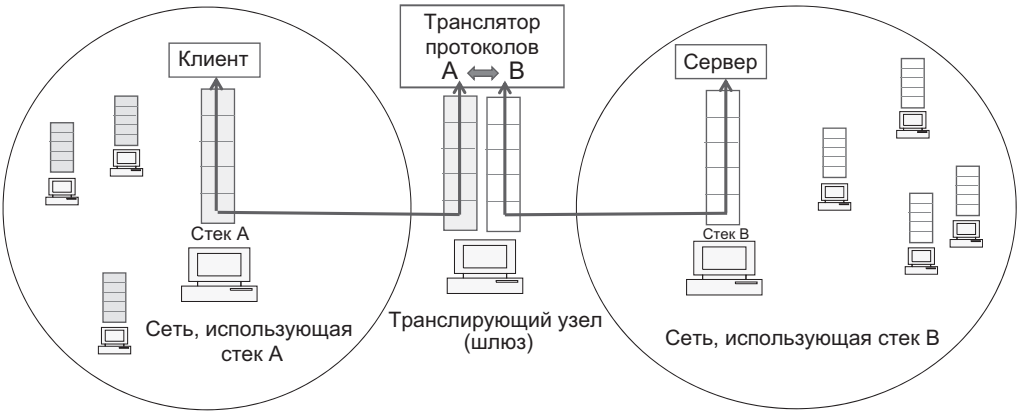


Рис. 17.15. Схема трансляции стеков протоколов

Инкапсуляция, или **туннелирование**, — еще один метод решения задачи согласования сетей, который, однако, применим только для согласования *транспортных* протоколов и лишь тогда, когда узлы двух сетей с одной транспортной технологией необходимо соединить через *транзитную* сеть с другой транспортной технологией. В процессе инкапсуляции принимают участие:

- *протокол-пассажир* — транспортный протокол объединяемых сетей;
- *несущий протокол* — транспортный протокол транзитной сети;
- *протокол инкапсуляции* — протокол, с помощью которого пакеты протокола-пассажира помещаются в поле данных пакетов несущего протокола.

Пакеты протокола-пассажира никоим образом не обрабатываются при транспортировке по транзитной сети. Инкапсуляцию выполняет пограничное устройство (маршрутизатор или шлюз), оснащенное двумя стеками протоколов, которое располагается на границе между исходной и транзитной сетями. Извлечение пакетов-пассажиров из несущих пакетов выполняет второе пограничное устройство, находящееся на границе между транзитной сетью и сетью назначения. Пограничные устройства указывают в несущих пакетах свои адреса, а не адреса из пакетов-пассажиров. Недостаток способа заключается в том, что узлы связываемых сетей не имеют возможности взаимодействовать с узлами транзитной сети.

Заметим, что роль пограничного устройства может выполнять хост, и в таком случае мы имеем еще две схемы организации туннеля: «хост — хост» и «хост — пограничное устройство». Туннельный режим передачи данных используется в самых разных сетевых технологиях не только для того, чтобы согласовать транспортные протоколы, но и для других целей. В качестве пары «несущий протокол — протокол-пассажир» могут высту-

пать транспортные протоколы как одного уровня (например, Ethernet инкапсулируются в Ethernet или IP в IP), так и разных уровней (PPP в IP). Например, при передаче группового трафика IP-пакеты с групповыми адресами упаковываются в обычные IP-пакеты и в таком виде пересылаются между маршрутизаторами, поддерживающими протокол IGMP. В этом случае туннели IP-Over-IP позволяют преодолеть те части сети, в которых не поддерживается групповое вещание. Другой пример — протокол IPSec (см. главу 31), где туннельный режим используется для транспортировки зашифрованных пакетов IP, инкапсулированных в обычные пакеты IP через незащищенные общедоступные сети.

Способы сосуществования сетей IPv4 и IPv6

В условиях параллельного существования протоколов IPv6 и IPv4 возможны следующие стратегии:[1]

- ничего не менять, сохранять однородное IPv4-окружение;
- строить чистую IPv6-сеть;
- мигрировать в сторону IPv6 с частичным сохранением IPv4.

Начнем с первой стратегии — *ничего не менять*. Действительно, нет ничего, что бы принципиально исключало такую стратегию. Среди самых первых требований к IPv6 разработчики называли возможность сетей IPv4 сосуществовать с IPv6 неопределенно долго. Но проблема дефицита адресов является объективной реальностью и со временем не исчезает, а лишь обостряется. На рис. 17.16 представлены данные об исчерпании централизованно распределяемых адресов IPv4.

Верхняя горизонтальная линия здесь соответствует пулу адресов, равному числу узлов в сети класса A. Как видно из рисунка, с середины 2017 года такого количества адресов не имеет ни одна региональная организация. В этих условиях обладатели сетей IPv4 должны полагаться на *технологию трансляции адресов NAT* и на все еще имеющиеся на рынке адреса IPv4. Кроме того, они должны быть готовы к неизбежному возникновению проблем при доступе к чужим узлам, на которых установлены стеки IPv6.

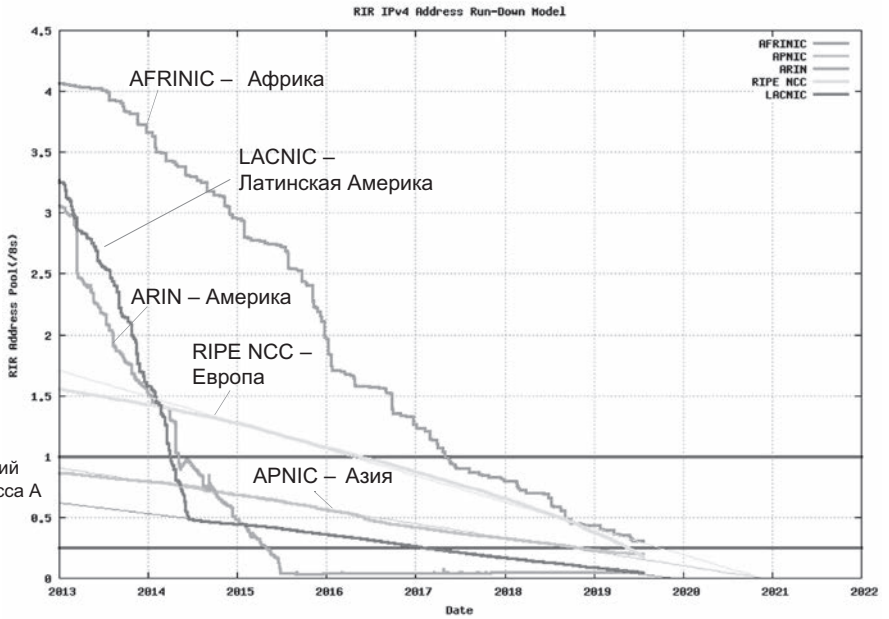
Другая, прямо противоположная стратегия — это *строительство чистой IPv6-сети*.

> **Чистая** (native) **IPv6-сеть**, называемая также сетью **IPv6-only**, — это сеть, построенная исключительно на стеке IPv6, в которой вся сетевая инфраструктура: операционные системы всех компьютеров, сетевые службы (DHCP и DNS), средства маршрутизации, системы мониторинга и безопасности — модернизирована так, чтобы постоянно и устойчиво поддерживать IPv6.

Хотя чистая IPv6-сеть является конечной целью многих организаций, в настоящее время таких сетей относительно немного, поскольку для их построения требуются немалые дополнительные затраты на модернизацию не совместимых с IPv6 устройств и приложений, которых в любой сети обычно имеется достаточно много. Идеальный путь к сети IPv6-only — это построение сети с *чистого листа* для вновь создаваемых сайтов, использующих самое современное оборудование и приложения.

---

[1] Заметим, что выбор той или иной позиции не может быть сделан в отрыве от конкретных бизнес-обстоятельств, специфики имеющейся инфраструктуры, наличия ресурсов и т. д.



Пул адресов, соответствующий одной сети класса A

Рис. 17.16. Модель исчерпания распределяемых централизованно адресов IPv4 (источник http://ipv4.potaroo.net/, 20 июля 2019 года)

И наконец, наиболее взвешенной стратегией многие эксперты считают планомерное *развертывание IPv6 с сохранением наследия IPv4*, хотя и в этом случае практически невозможно обойтись без сложного переходного процесса, включающего различные этапы согласования технологий IPv4 и IPv6. Существует широкий набор различных методов согласования, но все они базируются на рассмотренных трех основных подходах: двойном стеке, туннелировании и трансляции.

Двойной стек (dual stack). **Наиболее естественным и универсальным методом интеграции сети с глобальным IPv6-интернетом является установка стека IPv6 на каждый узел сети с последующим подключением ее к IPv6-сети провайдера. Обычно при внедрении IPv6 организации не отказываются полностью от IPv4, и в результате хосты и маршрутизаторы их сетей поддерживают *оба* стека протоколов (рис. 17.17).**

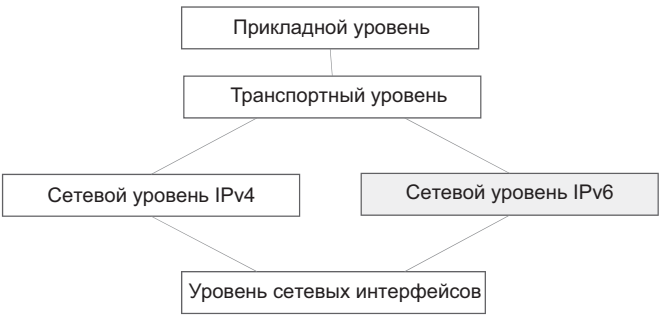


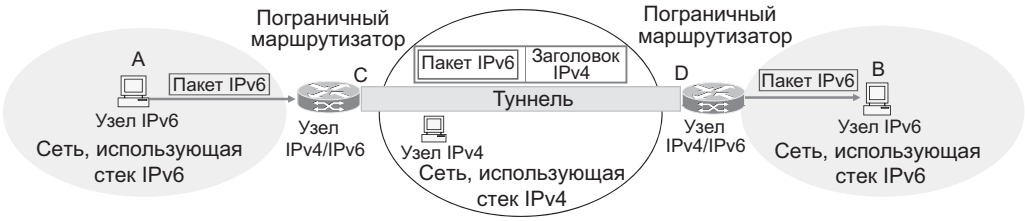
Рис. 17.17. Двойной стек для узлов IPv4/IPv6

Каждому такому узлу, называемому **узлом IPv4/IPv6**, назначается два набора конфигурационных параметров (адреса интерфейса, маршрутизатора по умолчанию, DNS-серверы, параметры DHCP и т. д.): один для стека IPv4, второй — для IPv6. Для каждого стека используется соответствующая система адресации. В том случае, когда IPv6-хост отправляет сообщение IPv6-хосту, он использует стек IPv6, а если тот же хост взаимодействует с IPv4-хостом — стек IPv4. Как видим, данный метод не содержит никакого специального транзитного механизма. Но он требует специального внимания к конфигурированию внешней и внутренней маршрутизации, так как не все протоколы маршрутизации IPv4 могут работать в сети IPv6.

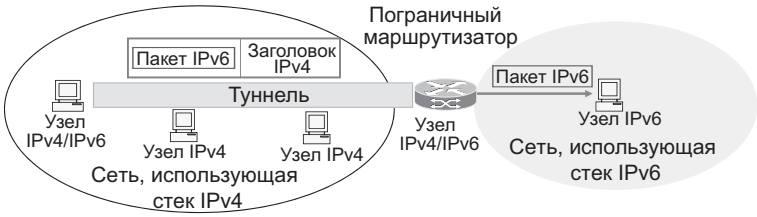
В настоящее время практически все операционные системы, а также многие приложения и сетевые службы оснащены полнофункциональными версиями стеков IPv4 и IPv6. Таким образом, схема «двойной стек» стала основной при интеграции сетей. Но означает ли это, что пользователь может в любой момент перейти на использование стека IPv6? Рассмотрим в качестве примера работу веб-браузера Firefox. Когда пользователь обращается к некоторому сайту, браузер делает запросы типа A и AAAA к DNS-серверу и получает в ответ два адреса, IPv4 и IPv6. Далее запускается процедура выбора предпочтительного адреса, а следовательно, и предпочтительного стека. Во многих браузерах, в том числе и браузере Firefox, этот выбор выполняется на основе алгоритма **Happy Eyeballs** (RFC 6555). В соответствии с этим алгоритмом браузер делает два запроса по обоим адресам, а затем сравнивает время ответа. Для использования выбирается тот стек, который дал более короткое время. Очевидно, что время ответа зависит от того, насколько окружающая сетевая инфраструктура поддерживает тот или иной стек. Кроме того, в большинстве браузеров имеются настройки, которые позволяют пользователю сконфигурировать его для работы исключительно по IPv4 или по IPv6.

Туннелирование может быть применено в тех случаях, когда две сети IPv6 необходимо соединить через транзитную сеть IPv4 (или наоборот). Этот метод часто используется большими компаниями, которые не хотят тратить слишком много времени и средств, чтобы перевести их большие сети полностью на IPv6 или поддерживать на всех узлах двойной стек. Однако туннелирование становится слишком громоздким и дорогостоящим методом с ростом числа сетей, которые требуется соединять друг с другом с помощью туннелей (особенно это справедливо для топологий, близких к полносвязной). На рис. 17.18 показаны схемы двух разновидностей туннелей: маршрутизатор — маршрутизатор и хост — маршрутизатор.

В первом случае узел IPv6 с адресом A посылает пакет другому узлу IPv6, имеющему адрес B. Данный пакет продвигается маршрутизаторами сети IPv6 на основании адреса назначения B и достигает промежуточного маршрутизатора, который выполняет функции пограничного устройства туннеля в IPv4-сети. Маршрутизатор формирует IPv4-пакет, в поле данных которого упаковывает данный IPv6-пакет, а в полях адреса источника и назначения заголовка IPv4-пакета указывает IPv4-адреса C и D соответственно. На основании адреса назначения D пакет IPv4, несущий пакет IPv6, продвигается по сети IPv4 абсолютно таким же образом, как и все остальные пакеты этой IPv4-сети, пока не достигнет адреса назначения D — второго пограничного маршрутизатора. Маршрутизатор, получив пакет, анализирует его заголовок и определяет, что в поле «Протокол верхнего уровня» содержится значение 41, указывающее на то, что в поле данных этого пакета находится пакет IPv6. Пограничный маршрутизатор извлекает исходный IPv6-пакет и в неизменном виде маршрутизирует его обычным образом на основании IPv6-адреса назначения B.



а) Туннель маршрутизатор — маршрутизатор



б) Туннель хост — маршрутизатор

Рис. 17.18. Согласование технологий IPv4 и IPv6 путем туннелирования

В схеме хост — маршрутизатор узел IPV4/IPv6, расположенный в сети с преобладанием узлов IPv4-only, выполняя роль пограничного устройства, создает аналогично работающий туннель до другого пограничного устройства (маршрутизатора).

Разные варианты туннелирования отличаются способом конфигурирования туннелей. Например, метод GRE (Generic Routing Encapsulation) использует ручное конфигурирование пограничных устройств туннеля, а метод 6to4 — автоматическое конфигурирование с привлечением особого типа IPv6-адресов с префиксом 2002::/16.

Трансляция протоколов используется, если требуется обеспечить взаимодействие узла сети IPv6-only и узла IPv4-only. Согласование двух версий протокола IP происходит путем преобразования заголовков IPv4 ⇔ IPv6. Процесс преобразования включает, в частности, отображение адресов сетей и узлов, различным образом трактуемых в этих протоколах. Отображение адресов часто выполняется на основе различных вариантов технологии NAT[1]. Наиболее популярна технология трансляции **NAT64** — рассмотрим ее работу на примере, приведенном в RFC 6146. Основным транслирующим элементом сети является шлюз NAT64 с двумя интерфейсами, один из которых соединен с сетью IPv4, а другой — с сетью IPv6 (рис. 17.19).

Трафик из сети IPv6 в сеть IPv4 проходит через этот шлюз, который выполняет трансляцию каждого проходящего пакета в соответствии с алгоритмом, приведенным в RFC 6145, преобразуя структуру и содержимое заголовка IPv6 в заголовок IPv4, в том числе отображая адреса IPv6 на IPv4. В данном методе шлюз может работать как в одном, так и в другом

[1] Так, в главе 30 описывается, как пограничное устройство NAT отображает множество частных IP-адресов внутренней сети на несколько публичных IP-адресов с целью уменьшить количество используемых предприятием публичных адресов или же для того, чтобы в целях безопасности скрыть внутреннюю структуру сети. Тот же самый механизм используется и для отображения адресов IPv6 на IPv4.

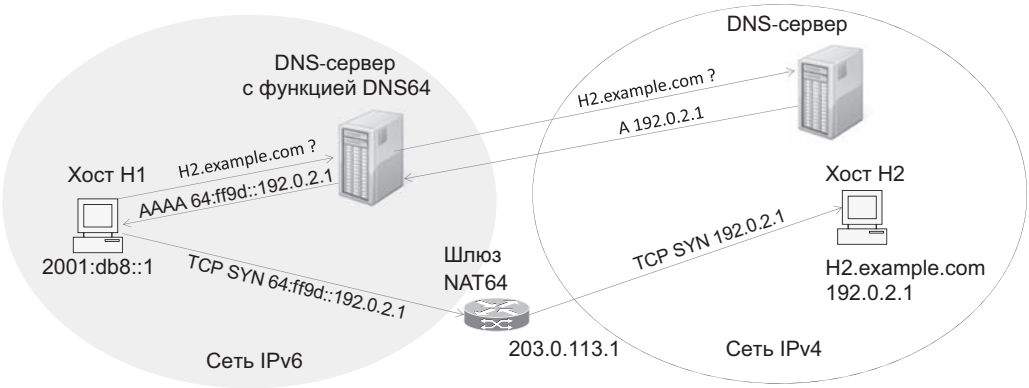


Рис. 17.19. Трансляция протоколов IPv6 ⇔ IPv4 с использованием технологии NAT64

направлении, однако он не является симметричным: поскольку адресное пространство IPv6 намного больше адресного пространства IPv4, однозначное отображение адресов IPv6 в IPv4 невозможно. Чтобы выполнять трансляцию IPv6 в IPv4, шлюз NAT64 должен запоминать соответствие между адресом и портом (IPv6, порт) хоста с одной стороны и адресом и портом (IPv4, порт) шлюза – с другой[1]. Пару (IP-адрес, порт) будем называть *транспортным адресом*. Для работы шлюза NAT64 нужны по крайней мере один адрес IPv4 и блок адресов IPv6, состоящий из 32-битного адресного пространства. На рисунке показаны:

- Хост H1 с адресом IPv6 2001:db8::1.
- Хост H2 с адресом IPv4 192.0.2.1 и именем h2.example.com.
- Шлюз NAT64 имеет интерфейс в сети IPv4 с адресом 203.0.113.1. Шлюзу назначен стандартизованный (well known) префикс 64:ff9b::/96, используемый для представления адресов IPv4 в адресном пространстве IPv6.
- Локальный DNS-сервер с функцией DNS64, заключающейся в том, что сервер использует префикс 64:ff9b::/96 для создания адресов IPv6 из адресов IPv4.

Когда хост H1 хочет установить связь с хостом H2, происходит следующее:

1. Хост H1 выполняет DNS-запрос с именем h2.example.com и получает ответ типа AAAA от локального DNS-сервера, например 64:ff9b::192.0.2.1.
2. Хост H1 посылает TCP SYN-пакет хосту H2. Пакет посылается с транспортным адресом источника (2001:db8::1, 1500) по транспортному адресу назначения (64:ff9b::192.0.2.1, 80), где номера портов 1500 и 80 выбираются хостом H1.
3. Таблица маршрутизации сети IPv6 сконфигурирована таким образом, что данный пакет, посланный хостом H1, маршрутизируется к интерфейсу IPv6 шлюза NAT64.
4. Шлюз NAT64 получает пакет и выполняет следующие действия:
   - выбирает первый же неиспользуемый им порт, например 2000, для своего IPv4-адреса 2003.0.113.1 и создает отображение (2001:db8::1,1500) ⇔ (2003.0.113.1, 2000);

[1] Такой подход используется в технологии NAPT (см. главу 30).

- шлюз NAT64 транслирует заголовок IPv6 в IPv4, в результате содержимое полей заголовка или заголовков IPv6 преобразуется в содержимое полей заголовка IPv4;
- шлюз NAT64 использует (2003.0.113.1, 2000) как транспортный адрес отправителя пакета, а (192.0.2.1, 80) как транспортный адрес назначения. При этом 192.0.2.1 непосредственно извлекается из поля IPv6-адреса назначения 64:ff9b::192.0.2.1 пришедшего пакета. Порт назначения 80 транслируемого пакета остается тем же, то есть переносится без изменения из транспортного адреса назначения.

5. После трансляции шлюз посылает пакет в сеть IPv4 со своего интерфейса IPv4.
6. Хост H2 получает пакет и отвечает пакетом TCP SYN+ACK с транспортным адресом назначения (2003.0.113.1, 2000) и транспортным адресом отправителя (192.0.2.1, 80).
7. По IPv4-адресу 2003.0.113.1 пакет поступает в шлюз NAT64, шлюз проверяет, нет ли среди имеющихся у него отображений таких, которые бы содержали пару (2003.0.113.1, 2000). Так как отображение (2001:db8::1,1500) ⇔ (2003.0.113.1, 2000) существует, то шлюз выполняет следующие действия:
   - шлюз NAT64 выполняет обратную трансляцию заголовка IPv4 в IPv6;
   - шлюз NAT64 использует (2001:db8::1, 1500) как транспортный адрес назначения пакета и (64:ff9b::192.0.2.1, 80) как транспортный адрес источника пакета. При этом 192.0.2.1 непосредственно извлекается из поля адреса источника пришедшего пакета IPv4. Значение порта источника (80) переносится без изменения в пакет IPv6 из пакета IPv4.
8. Полученный в результате трансляции пакет посылается через интерфейс IPv6 хосту H1.

Завершая раздел, заметим, что при миграции в сторону IPv6 владельцы и разработчики больших сетей редко ограничиваются одной технологией. Гораздо чаще наиболее подходящий способ интеграции выбирается для каждой подсети, а иногда и для каждого отдельного узла, что приводит к внедрению различных механизмов перехода от IPv4 к IPv6, отвечающих специфическим требованиям и учитывающих конкретные особенности разных частей сети.

Вопросы к части IV

Ответьте на вопросы 1–4, используя следующие пронумерованные IP-адреса:

(1) 123.1. 223.5	(2) 225.0.0.1	(3) 194.87.45.255	(4) 10.124.251.252	(5) 125.24.255.255
(6) 17.213.355.205	(7) 179.12.255.255	(8) 127.0.23.55	(9) 1.0.0.13	(10) 124.1.1.1
(11) 192.13.16.255	(12) 0.0.0.0	(13) 13.13.13.13	(14) 12.145.3.12.5	(15) 172.17. 1.4

1. Какие из адресов, приведенных выше, являются синтаксически ошибочными?
2. Какие адреса относятся к классу A? Классу B? Классу C?
3. Какие из адресов являются широковещательными адресами класса C?
4. Какие из адресов не могут быть назначены в качестве индивидуального глобального адреса интерфейса?
5. Пусть IP-адрес некоторого узла подсети равен 108.5.18.167, а значение маски для этой подсети — 255.255.255.240. Определите номер подсети. Какое максимальное число сетевых интерфейсов может быть в этой подсети?
6. Какое максимальное количество подсетей теоретически можно организовать, если в вашем распоряжении имеется сеть класса C? Какое значение должна при этом иметь маска? При ответе не принимайте во внимание двухточечные соединения.
7. Пусть вам ничего не известно об узлах, кроме их DNS-имен: w1.piter.ru, www.msk.ru и www.piter.ru, echo.msk.ru. Что вы можете сказать о том, насколько близко территориально находятся они относительно друг друга? Варианты ответов:

   а) узел www.msk.ru территориально ближе к echo.msk.ru, чем к w1.piter.ru;

   б) узел www.msk.ru расположен ближе к www.piter.ru, чем к echo.msk.ru;

   в) узел w1.piter.ru территориально ближе к www.piter.ru, чем к www.msk.ru;

   г) ничего определенного.
8. Пусть вам ничего не известно о структуре сети, но в вашем распоряжении имеется следующая таблица соответствия IP-адресов и DNS-имен нескольких узлов сети:

IP-адрес узла	123.1.0.01	123.1.0.02	123.1.0.03	123.1.0.04	?	?
DNS-имя узла	w1.mgu.ru	w2.mgu.ru	w3.mgu.ru	w4.mgu.ru	w5.mgu.ru	w6.mgu.ru

   Что вы можете сказать об IP-адресах узлов, имеющих DNS-имена w5.mgu.ru и w6.mgu.ru?
9. Какая запись из следующих эквивалентна значению маски /29 (выберите вариант ответа):

   а) 255.255.2.9;

   б) 255.255.255.29;

   в) 255.255.255.248;

   г) 255.255.255.229.

10. Могут ли для двухточечной связи быть назначены адреса 197.220.12.9/31 и 197.220.12.10/31?
11. Содержимое ARP-таблицы изменяется в результате следующих событий (выберите вариант ответа):
    а) при получении ARP-ответов на запрос;
    б) при получении широковещательных ARP-запросов;
    в) по истечении времени жизни записи в ARP-таблице.
12. Протокол ARP функционально можно разделить на клиентскую и серверную части. Опишите, какие функции вы отнесли бы к клиентской части, а какие — к серверной?
13. В чем состоят функции нижнего уровня стека TCP/IP (выберите вариант ответа):
    а) кодирование и мультиплексирование электрических сигналов;
    б) инкапсуляция и декапсуляция данных сетевого уровня в кадры нижележащей технологии;
    в) формирование кадров и синхронизация;
    г) доступ к среде передачи;
    д) преобразование сетевых адресов в адреса нижележащей технологии.
14. Поясните, что понимается под термином «линия связи» (Link) в технологии TCP/IP (выберите вариант ответа):
    а) любая коммуникационная среда, которая позволяет IP-узлам взаимодействовать друг с другом без промежуточных маршрутизаторов;
    б) физическая линия связи (отрезок кабеля) с работающим на ней протоколом канального уровня;
    в) локальная сеть, построенная на коммутаторах, связывающая интерфейсы маршрутизаторов;
    г) отдельный коммутируемый метками путь MPLS.
15. Передается ли в IP-пакете маска в тех случаях, когда маршрутизация реализуется с использованием масок?
16. Какие элементы сети могут выполнять фрагментацию? Варианты ответов:
    а) только компьютеры;
    б) только маршрутизаторы;
    в) компьютеры, маршрутизаторы, мосты, коммутаторы;
    г) компьютеры и маршрутизаторы.
17. Что произойдет, если при передаче пакета он был фрагментирован, причем один из фрагментов не дошел до узла назначения по истечении тайм-аута? Варианты ответов:
    а) IP-модуль узла-получателя отбросит все полученные фрагменты пакета, в котором потерялся один фрагмент, а IP-модуль узла-отправителя не будет предпринимать никаких действий по повторной передаче данного пакета;
    б) IP-модуль получателя сообщит о неполучении одного фрагмента, а IP-модуль узла-отправителя повторит передачу недошедшего фрагмента;

в) IP-модуль получателя сообщит о неполучении одного фрагмента, а IP-модуль узла-отправителя повторит передачу всего пакета, в состав которого входил недошедший фрагмент.

18. Кому адресовано ICMP-сообщение? Варианты ответов:
    а) протоколу IP узла-отправителя пакета, вызвавшего ошибку;
    б) протоколу транспортного или прикладного уровня узла-отправителя пакета, вызвавшего ошибку;
    в) протоколу IP ближайшего маршрутизатора, от которого поступил пакет, вызвавший ошибку.
19. Сколько ARP-таблиц имеет компьютер? маршрутизатор? коммутатор?
20. В студенческом общежитии живет 350 студентов, и каждый из них имеет собственный ноутбук. В общежитии оборудована специальная комната, в которой развернута компьютерная сеть, имеющая 20 коннекторов для подключения компьютеров. Время от времени студенты работают в этом компьютерном классе, подключая свои ноутбуки к сети. Каким количеством IP-адресов должен располагать администратор этой компьютерной сети, чтобы все студенты могли подключаться к сети, не выполняя процедуру конфигурирования своих ноутбуков при каждом посещении компьютерного класса?
21. Какие из следующих утверждений верны:
    а) broadcast является частным случаем multicast;
    б) broadcast является частным случаем anycast;
    в) multicast является частным случаем anycast;
    г) ни один из этих способов рассылки не является частным случаем другого.
22. Почему провайдеры при выделении адресов своим клиентам стремятся к тому, чтобы адресные пространства сетей, располагающихся территориально по соседству, имели совпадающие префиксы?
23. Какие из следующих утверждений правильны:
    а) клиентом DNS-сервера является резольвер;
    б) DNS-клиентом является практически каждый узел интернета;
    в) запись типа AAAA файла зоны отображает DNS-имя в IPv4-адрес;
    г) для каждой зоны существует один первичный и несколько вторичных DNS-серверов.
24. Сравните таблицу моста или коммутатора с таблицей маршрутизатора. Каким образом формируются эти таблицы? Какую информацию содержат? От чего зависит их объем?
25. Рассмотрим маршрутизатор на магистрали интернета. Какие записи содержатся в поле адреса назначения его таблицы маршрутизации? Варианты ответов:
    а) номера всех сетей интернета;
    б) номера некоторых сетей интернета;
    в) номера некоторых сетей и адреса некоторых конечных узлов интернета;
    г) номера сетей, подсоединенных к интерфейсам данного маршрутизатора.

26. Какой маршрут к определенному адресу назначения в таблице маршрутизации считается более специфическим? Какому маршруту — более специфическому или менее специфическому — отдается предпочтение при выборе маршрута?
27. Пусть префикс непрерывного пула IP-адресов составляет 12 двоичных разрядов. Сколько адресов входит в этот пул? Варианты ответов:
    а) 2^{12};
    б) 2^{20};
    в) $2^{12} - 2$;
    г) 12^2.
28. Передается ли в IP-пакете маска в тех случаях, когда маршрутизация реализуется с использованием масок?
29. Приведите примеры, когда может возникнуть необходимость в использовании специфических маршрутов.
30. Какие преимущества дает технология CIDR? Что мешает ее широкому внедрению?
31. Почему в записи о маршруте по умолчанию в качестве адреса сети назначения часто указывается 0.0.0.0 с маской 0.0.0.0?
32. В каких случаях система DNS использует протокол UDP, а в каких — TCP?
33. Если при обмене данными по методу с возвращением на N пакетов отправитель получил квитанцию на (n + 1)-й пакет, а квитанция на предыдущий n-й пакет не пришла, то (выберите вариант ответа):
    а) отправитель считает n-й пакет успешно принятым и продолжает передачу;
    б) по истечении тайм-аута отправитель повторно отсылает n-й пакет;
    в) по истечении тайм-аута получатель повторно отсылает квитанцию на n-й пакет.
34. Как соотносятся размеры окна приема и окна передачи в методе с выборочным повторением (выберите вариант ответа):
    а) окно передачи больше, чем окно приема;
    б) они равны;
    в) окно приема больше, чем окно передачи.
35. Может ли приложение, используя протокол UDP, обеспечить надежную связь?
36. Какой объем данных получен в течение TCP-сеанса отправителем TCP-сегмента, в заголовке которого в поле квитанции помещено значение 160005, если известно, что первый полученный байт имел номер 15000?
37. Укажите, в каком виде передаются квитанции на получение сегментов в протоколе TCP (выберите вариант ответа):
    а) квитанция передается в поле данных TCP-сегмента с установленным в заголовке флагом ACK;
    б) квитанция — это флаг ACK;
    в) квитанция — это значение поля последовательного номера в заголовке TCP-сегмента с установленным флагом ACK;

г) квитанция — это значение поля подтвержденного номера в заголовке TCP-сегмента с установленным флагом ACK.

38. Проведите с партнером сеанс моделирования работы протокола TCP. Для этого договоритесь с ним о максимальном размере сегмента, о начальных размерах буферов, о начальном значении порядкового номера, о размерах окна. Затем асинхронно начните «посылать» друг другу «сегменты» — карточки, на которых заполнены ключевые поля: номер первого байта, размер посылаемого сегмента, номер квитанции и, если требуется, новое значение размера окна. Время от времени «теряйте» карточки при передаче и выполняйте действия, соответствующие логике TCP. Не забудьте делать временные отметки на каждой копии отправленного сегмента, чтобы отслеживать приход квитанций. Выполнение этого задания не только сделает модуль TCP более понятным для вас, но и, что гораздо важнее, породит новые вопросы.

39. Какой параметр протокола TCP используется для регулирования: а) заполнения буфера приемника; б) загруженности сети? Какая сторона — отправитель или получатель — устанавливает каждый из этих параметров?

40. Может ли работать маршрутизатор, не имея таблицы маршрутизации? Варианты ответов:

    а) может, если выполняется маршрутизация от источника;

    б) нет, это невозможно;

    в) может, если выполняется лавинная маршрутизация;

    г) может, если в маршрутизаторе задан маршрут по умолчанию.

41. К какому типу относится протокол OSPF? Варианты ответов:

    а) протокол маршрутизации;

    б) протокол, основанный на алгоритме состояния связей;

    в) адаптивный протокол;

    г) централизованный;

    д) дистанционно-векторный;

    е) проактивный.

42. Какие параметры сети учитывают метрики, поддерживаемые протоколом OSPF? Варианты ответов:

    а) пропускная способность;

    б) время передачи;

    в) надежность каналов связи;

    г) количество хопов.

43. Поясните, какие недостатки традиционной маршрутизации привели к созданию SDN (выберите вариант ответа):

    а) негибкая процедура продвижения трафика на основе единственного поля — адреса назначения;

    б) децентрализованные протоколы построения таблиц маршрутизации медленно реагируют на изменения топологии сети;

в) децентрализованные протоколы построения таблиц маршрутизации позволяют находить только квазиоптимальные маршруты;

г) для агрегированных потоков на интернет-магистралях необходима гибкая и адаптивная обработка на основе логического разделения трафика на отдельные потоки;

д) нестандартный командный язык конфигурирования сетевых устройств препятствует унификации управления этими устройствами.

44. Система, в которой функции сетевых устройств — маршрутизаторов, коммутаторов, файерволов и др. — реализуются не на основе аппаратной специализированной платформы, а программным путем в серверах общего назначения, называется (выберите вариант ответа):

а) виртуальной частной сетью VPN;

б) виртуальной локальной сетью VLAN;

в) виртуальной сетью NFV.

45. Какой MAC-адрес имеет сетевой интерфейс, если известно, что его глобальный уникальный адрес 2001:718::280:48FF:FEEB:7960 был получен в результате автоконфигурирования?

46. Какие из записей IPv6-адреса FEDC:0000:0000:0006:0030:0000:0000:7654 являются синтаксически ошибочными (выберите вариант ответа):

а) FEDC:: 6: 30:0:0:7654;

б) FEDC::0006:30:0:0:7654;

в) FEDC:: 6:30::7654;

г) FEDC:0:0: 6:3::7654.

47. Какие типы адресов определены в IPv6 (выберите вариант ответа):

а) anycast;

б) multicast;

в) broadcast;

г) unicast;

д) loopback.

48. Объясните назначение группового адреса запрашиваемого узла (Solicited-Node Multicast Address). Почему в группу, описываемую этим адресом, скорее всего, входит только один узел?

49. Поле «Класс трафика» в IPv6 аналогично полю «Тип сервиса» в IPv4 и используется для организации обслуживания с различным уровнем QoS. Зачем в IPv6 введено еще одно поле — «Метка потока», также направленное на поддержание QoS? Какие дополнительные возможности оно предоставляет?

50. Чем отличается информация в основном заголовке IPv6 от информации заголовка IPv4 (выберите вариант ответа):

а) в IPv6 отсутствует поле «Контрольная сумма»;

б) в IPV6 появилось новое поле «Метка потока»;

в) поле «Длина поля данных» имеет большую разрядность, достаточную для задания длины джамбограммы;

г) поля «Адрес источника» и «Адрес назначения» имеют большую разрядность;

д) основной заголовок IPv6 короче заголовка IPv4.

51. Какие из перечисленных технологий наиболее подходят, когда узлы двух сетей IPv6 необходимо соединить через транзитную сеть IPv4 (выберите вариант ответа):

а) транслирование стеков;

б) инкапсуляция IPv4 in IPv6;

в) туннелирование IPv6 over IPv4;

г) мультиплексирование стеков;

д) двойной стек.

52. Предложите несколько возможных стратегий поведения владельцев сетей в условиях существования двух разных стеков – IPv6 и IPv4.

53. На веб-сервере и на вашем клиентском компьютере установлен двойной стек: IPv4 и IPv6. Каким образом браузер на вашем компьютере может выбрать, по какому протоколу он будет взаимодействовать с этим веб-сервером?

54. Опишите процедуру проверки наличия дубликата адреса с помощью протокола ND (рис. 17.9) для случая, когда интерфейс узла А имеет MAC-адрес 0C-1C-09-48-EB-5C-65 и неподтвержденный индивидуальный адрес 2001:620:1234:F8::41. Все остальные параметры примера остаются прежними.

Часть V

Глобальные проводные компьютерные сети

- ❑ Глава 18. Организация и коммуникационные услуги глобальных сетей
- ❑ Глава 19. Информационные услуги глобальных сетей
- ❑ Глава 20 . MPLS — транспортная технология глобальных сетей
- ❑ Глава 21 . Виртуальные частные сети

Глобальные IP-сети имеют сложную структуру, включающую сети доступа, магистральную сеть и многочисленные центры данных. На основе этой структуры провайдеры оказывают разнообразные услуги, как информационные, так и коммуникационные. Информационные услуги — это такие услуги, при оказании которых провайдер принимает во внимание содержание передаваемой пользователю информации. Например, социальные сети, стриминговые сервисы, услуги по обеспечению безопасности, поисковые службы, ИИ-помощники. Коммуникационные услуги заключаются в предоставлении пользователю технической возможности передачи данных через сеть, при этом провайдер не вдается в смысл передаваемых данных. Примером являются услуги доступа к интернету.

Организация глобальных сетей, а также общий обзор услут, оказываемых ими, рассматриваются в главе 18.

Глава 19 посвящена информационным услугам, а именно веб-службе, электронной почте и облачным сервисам. В этой главе также рассматриваются услуги управления сетью, которые могут оказываться как провайдером услуг, так и службой технической поддержки корпоративной сети.

Коммуникационные услуги изучаются в двух последующих главах.

Основным транспортным средством, на основе которого строятся глобальные сети, является IP-технология. Типичная глобальная сеть имеет многоуровневую структуру, в которой IP занимает верхний уровень (если рассматриватьтолькоуровни, обеспечивающие транспортировкуданных), а подуровнем IP работают другие пакетные технологии. Спецификой глобальных сетей является то, что под уровнем IP часто применяются технологии, основанные на технике виртуальных каналов. Причина популярности технологий этоготипа в том, что они обеспечивают гораздо более высокую степень контроля над соединениями между пользователями сети и путями прохождения информационных потоков через узлы сети, чем дейтаграммная техника, и это свойство очень привлекательно для оператора глобальной сети и провайдера услуг. Опыт сосуществования IP с технологиями, основанными на механизме виртуальных каналов, привел в середине 90-х к по-

явлению гибридной технологии MPLS, столь тесно интегрированной с протоколами стека IP, что иногда ее называют IP/MPLS,

В главе 20 изучаются основы техники виртуальных каналов и технология MPLS, на основе которой провайдер оказываетсегодня большой спектр коммуникационныхуслуг, например, с гарантированным качеством обслуживания с помощьюсегментной маршрутизации.

Одной из наиболее популярных услуг, оказываемых на основе MPLS, является услуга виртуальных частныхсетей (VPN), позволяющая объединитьотдельныетерриториально рассредоточенные сети некоторого предприятия в единую корпоративную сеть с помощью интернета. Виртуальная частная сеть имитирует свойства частной сети: изолированность от сетей других пользователей, предсказуемую производительность и безопасность,

В главе 21 рассматриваются общие принципы построения виртуальных частных сетей VPN, а также MPLSVPN.

ГЛАВА 18 Организация и коммуникационные услуги глобальных сетей

Сети операторов связи

Сегодня глобальные компьютерные сети операторов связи являются движущей силой и местом приложения практически всех новых транспортных технологий компьютерных сетей. Глобальные компьютерные сети значительно изменились со времени своего появления в конце 60-х. Одним из показателей этого прогресса служит изменение скорости передачи данных транспортными сетевыми технологиями глобальных сетей:

- 1980-е: магистраль интернета построена на цифровых телефонных каналах 56 Кбит/с; магистрали телефонных сетей используют цифровые линии 35–45 Мбит/с.
- 1990-е: в интернете начинают применяться магистрали SDH 155 и 622 Мбит/с.
- Конец 1990-х — начало 2000-х: иерархия скоростей SDH повышается до 10 Гбит/с; технология DWDM позволяет мультиплексировать в одном оптическом волокне до 40–80 каналов по 10 Гбит/с (общая пропускная способность волокна составляет 400–800 Гбит/с).
- Начало 2010-х: стандартизован 100G Ethernet, который начинает применяться на магистралях сетей OTN. Общая пропускная способность одного волокна повышается до 4–8 Тбит/с.
- Конец 2010-х: стандартизованы версии 200G и 400G Ethernet, начинают внедряться OTN, передающие сигналы 400G Ethernet на одной волне, пропускная способность одного волокна повышается до 10–30 Тбит/с.
- Начало 2020-х: стандартизован 800G Ethernet, идет работа над стандартизацией 1600G Ethernet.

Прогресс, впрочем, произошел не только в области высокоскоростной передачи данных — изменились и структура глобальных сетей, и их услуги.

Специализированное предприятие, создающее телекоммуникационную сеть для оказания общедоступных услуг, владеющее этой сетью и поддерживающее ее работу, называется **оператором связи** (telecommunication carrier). Если эта сеть предоставляет определенные услуги конечным пользователям, то оператор связи одновременно является и **провайдером услуг**, например, провайдер услуг доступа к интернету, провайдер услуг мобильной связи. Провайдер услуг может иметь свою собственную телекоммуникационную сеть, а может и арендовать часть инфраструктуры сети у некоторого оператора связи, чтобы оказывать свои услуги.

Глобальные компьютерные сети операторов связи превратились сегодня в телекоммуникационные сети, работающие на основе протокола IP. С их помощью операторы связи оказывают широкий спектр услуг как конечным пользователям, так и друг другу.

Операторы связи отличаются друг от друга:

- набором предоставляемых услуг;
- территорией, в пределах которой предоставляются услуги;
- типом клиентов, на которых ориентированы их услуги;
- имеющейся во владении оператора инфраструктурой — линиями связи, коммутационным оборудованием, информационными серверами и т. п.;
- отношением к монополии на предоставление услуг.

Услуги операторов связи

Современные операторы связи обычно оказывают услуги нескольких типов.

Услуги предоставления каналов связи в аренду относятся к самому нижнему уровню, так как пользователю приходится при этом самостоятельно выполнять дополнительную работу — строить с помощью предоставленных каналов собственную сетевую инфраструктуру (устанавливать маршрутизаторы компьютерных сетей, базовые станции и шлюзы мобильных сотовых сетей). Обычно к таким услугам обращаются другие операторы связи, у которых для построения своей сети нет собственных каналов связи.

Более высокоуровневые услуги сегодня создаются на основе IP-сетей. Эти услуги можно разделить на коммуникационные и информационные.

К коммуникационным услугам относятся:

- услуги высокоскоростного доступа к интернету;
- услуги мобильной телефонии (и более общо — услуги мультимедийной связи, передающие голос, видео и текстовые сообщения);
- услуги корпоративной IP-телефонии;
- услуги виртуальных частных сетей;
- услуги облачных вычислений;
- услуги безопасности — файерволы, защита от DDoS-атак;
- услуги интернета вещей — объединение сетей датчиков, устройств умных домов, промышленных установок.

И этот список можно продолжать.

В этой главе мы более подробно остановимся на характеристике только двух типов коммуникационных услуг — услугах виртуальных частных сетей и облачных вычислений. Другие типы коммуникационных услуг рассматриваются в последующих главах вместе с изучением технологий, на которых они основаны.

К информационным услугам[1] относятся:

---

[1] Информационные услуги рассматриваются в главе 19, а услуги безопасности в главах VII части книги.

- социальные сети;
- электронная почта;
- ИИ-помощники;
- веб-сервисы;
- IP-телевидение

и некоторые другие.

Для оказания коммуникационных услуг операторы связи применяют различные транспортные технологии — IP, Ethernet, PON, MPLS, OTN, DWDM, работающие на разных уровнях стека протоколов сети, которые обладают различными свойствами и могут работать в различных сочетаниях. Оператор связи использует эти технологии как для создания своих сетей, так и для предоставления услуг своим клиентам. Соотношение понятий *технология* и *услуга* можно пояснить следующими утверждениями:

- Одна и та же технология может быть использована для предоставления различных услуг. Например, технология IP может служить как для доступа в интернет, так и для организации виртуальной частной сети.
- Одна и та же услуга может быть реализована на основе разных технологий. Например, виртуальную частную сеть можно построить на основе технологий IP, Ethernet и MPLS.
- Имеются такие услуги, которые можно предоставлять на основе только одной специфической технологии. Например, услугу выделенной волны можно предоставлять только на основе технологии DWDM.

Потребители услуг

Все множество клиентов — потребителей инфотелекоммуникационных услуг — можно разделить на два больших класса: массовые индивидуальные клиенты и корпоративные клиенты.

В первом случае местом потребления услуг выступает квартира или частный дом, а клиентами — жильцы, которым нужны прежде всего базовые услуги — телефонная связь, телевидение, радио, доступ в интернет. Для **массовых индивидуальных клиентов** очень важна экономичность услуги — низкая месячная оплата, возможность использования стандартных терминальных устройств (телефонные аппараты, телевизионные приемники, персональные компьютеры), а также возможность задействовать существующую кабельную систему между ближайшим офисом оператора связи и домом клиента. Присутствующая во многих местностях традиционная телефонная проводка — серьезное ограничение для предоставления услуг доступа в интернет и новых услуг компьютерных сетей, так как она не была рассчитана на передачу данных, а подведение к каждому дому нового качественного кабеля, например волоконно-оптического, — дело дорогое (хотя этот вариант и становится все более доступным). Поэтому для предоставления компьютерных услуг таким клиентам разработаны специфические технологии доступа через существующие в доме окончания телефонной сети. В этом случае новые скоростные цифровые технологии доступа (DSL) используют телефонную сеть, но только на отрезке между домом клиента и офисом оператора связи, а далее данные передаются в обход телефонной сети по компьютерной сети с коммутацией пакетов. Существуют также технологии доступа, в которых для передачи данных применяется имеющаяся в городе сеть кабельного телевидения.

Корпоративные клиенты — это предприятия и организации различного профиля. Мелкие предприятия по набору требуемых услуг не слишком отличаются от массовых клиентов — это те же базовые телефония и телевидение, а также доступ к информационным ресурсам интернета. Крупные предприятия, состоящие из нескольких территориально рассредоточенных подразделений, а также имеющие сотрудников, часто работающих дома, нуждаются в расширенном наборе услуг и прежде всего в такой транспортной услуге, как *виртуальная частная сеть* (VPN), когда оператор связи создает для предприятия иллюзию того, что все его отделения и филиалы соединены частной сетью, то есть сетью, полностью принадлежащей предприятию-клиенту и полностью управляемой предприятием-клиентом (подробнее об этом см. главу 21 «Виртуальные частные сети»).

Корпоративные пользователи все чаще получают не только транспортные, но и информационные услуги операторов, например услуги хостинга, переносят собственные серверы, веб-сайты и базы данных на территорию оператора, поручая последнему поддерживать их работу и обеспечивать быстрый доступ к ним для сотрудников предприятия и, возможно, других пользователей сети оператора. Получившие в последнее время распространение облачные сервисы усилили эту тенденцию, позволяя корпоративным пользователям (и индивидуальным тоже) получать информационные услуги прозрачным способом, не заботясь об установке, конфигурировании и сопровождении серверов и программного обеспечения.

Инфраструктура

На формирование набора предлагаемых оператором услуг оказывает серьезное влияние их аппаратная база. Так, для оказания услуг по аренде каналов оператор должен иметь в своем распоряжении первичную сеть SDH/OTN/DWDM, а для оказания услуг виртуальных частных сетей — маршрутизаторы с функциональностью MPLS или коммутаторы Carrier Ethernet. Укрупненно типичная сеть оператора связи имеет двухслойную структуру, с нижним уровнем первичной сети, служащей фундаментом для глобальной IP-сети, на основе которой оказываются все современные услуги.

Структура глобальной IP-сети оператора связи в целом соответствует обобщенной структуре сети (см. раздел «Классификация компьютерных сетей» главы 4). Она состоит из магистральной сети, сетей агрегирования трафика и сетей доступа (рис. 18.1).

Коммуникационное оборудование пользователей взаимодействует с пограничным оборудованием сети доступа оператора связи по некоторому интерфейсу, называемому **интерфейсом «пользователь — сеть»** (User Network Interface, **UNI**). Для предоставления информационных компьютерных услуг в сети провайдера создаются **центры данных**. Центр данных — это локальная сеть с установленными серверами, на которых работает программное обеспечение, с помощью которого провайдеры предоставляют разнообразные услуги: веб-сервисы, IP-телефонию, облачные сервисы и т. п. В центрах данных могут также находиться серверы пользователей, если оператор предоставляет услуги *хостинга*.

Центры данных бывают магистральные и периферийные. *Магистральные центры данных* подключены к маршрутизаторам магистрали, а *периферийные центры данных* находятся в сетях доступа, в непосредственной близости от пользователей. Распределение в сетях доступа вычислительных ресурсов провайдера между магистральными центрами данных и периферийными центрами данных, находящимися на периферии сети, получило название **периферийных, или граничных, вычислений** (edge computing). Эта сравнительно

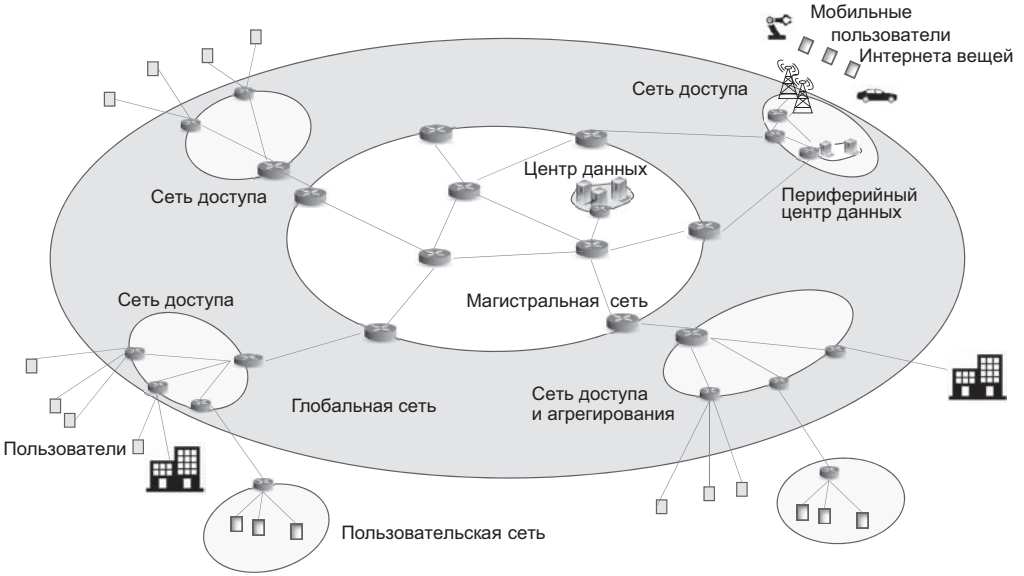


Рис. 18.1. Структура глобальной сети

новая тенденция отражает потребность пользователей нового типа — устройств интернета вещей — обмениваться данными с управляющими ими приложениями с минимальной задержкой, так как управление происходит в реальном масштабе времени. Поэтому такие приложения создаются распределенными, состоящими из нескольких частей. Части, выполняющие функции, требующие наиболее быстрой реакции на изменение состояния управляемого объекта, работают на серверах периферийных центров данных, а части, выполняющие фоновые функции, не требующие такой быстрой реакции, работают на серверах магистральных центров данных.

В тех случаях, когда у оператора отсутствует вся необходимая инфраструктура для оказания некоторой услуги, он может воспользоваться возможностями другого оператора; требуемая услуга может быть сконструирована на базе инфраструктуры партнера, а также собственной инфраструктуры. Примером такого рода является аренда оператором физических каналов связи для создания собственной IP-сети с тем, чтобы на ее основе оказывать услуги своим клиентам. Оператора, который предоставляет услуги другим операторам связи, часто называют **оператором операторов** (carrier of carriers).

Территория покрытия

По степени покрытия территории, на которой предоставляются услуги, операторы делятся на локальных, региональных, национальных и транснациональных.

Локальный оператор работает на территории городского или сельского поселения. Традиционный локальный оператор владеет всей соответствующей транспортной инфраструктурой: физическими каналами между помещениями абонентов (квартирами, домами и офисами) и центрами данных, в которых расположены коммутаторы, маршрутизаторы

и разнообразные серверы с приложениями, поддерживающие услуги локального провайдера.

Региональные и национальные операторы оказывают услуги на большой территории, располагая соответствующей транспортной инфраструктурой. Традиционные операторы этого масштаба выполняют транзитную передачу компьютерного и мультимедийного трафика между центрами данных локальных операторов, имея в своем распоряжении крупные центры данных и мощные маршрутизаторы, связанные высокоскоростными физическими каналами связи. Это — операторы операторов: их клиентами являются, как правило, локальные операторы или крупные предприятия, имеющие подразделения в различных городах региона или страны. Располагая развитой транспортной инфраструктурой, такие операторы обычно оказывают услуги дальней связи, передавая транзитом большие объемы информации без какой-либо обработки.

Транснациональные операторы оказывают услуги в нескольких странах. Они имеют собственные магистральные сети, покрывающие иногда несколько континентов. Часто подобные операторы тесно сотрудничают с национальными операторами, используя их сети доступа для доставки информации клиентам.

Взаимоотношения между операторами связи

Взаимосвязи между операторами различного типа (а также их сетями) иллюстрирует рис. 18.2, на котором показаны и индивидуальные, и корпоративные клиенты. Нужно иметь в виду, что каждый клиент обычно нуждается в услугах двух видов — телефонных и передачи данных. Индивидуальные клиенты имеют в своих домах или квартирах, как правило, телефон и компьютер, а у корпоративных клиентов имеются соответствующие сети — телефонная, поддерживаемая офисным телефонным коммутатором (PBX), и локальная сеть передачи данных, построенная на собственных коммутаторах.

Для подключения оборудования клиентов операторы связи организуют так называемые **точки присутствия** (Point Of Presents, **POP**) — здания или помещения, в которых размещается оборудование доступа, способное подключить большое количество каналов связи, идущих от клиентов. Иногда такую точку называют **центральным офисом** (Central Office, CO) — это традиционное название, унаследованное от операторов телефонных сетей. К POP локальных операторов подключаются абоненты, а к POP операторов верхних уровней — операторы нижних уровней или крупные корпоративные клиенты, которым необходимы высокие скорости доступа и большая территория покрытия, объединяющая их офисы и отделения в разных городах и странах.

Как видно из рисунка, в современном конкурентном телекоммуникационном мире нет строгой иерархии операторов, и взаимосвязи между ними и их сетями могут быть достаточно сложными и запутанными. Например, сеть локального оператора 5 имеет непосредственную связь не только с сетью регионального оператора 3, как того требует иерархия, но и непосредственную связь с национальным оператором 3 (возможно, этот оператор предлагает более дешевые услуги по передаче международного трафика, чем региональный оператор 3). Некоторые операторы могут не иметь собственной транспортной инфраструктуры (локальный оператор 1). Как это часто бывает в таких случаях, локальный оператор 1 предоставляет только дополнительные информационные услуги, например, предлагает клиентам локального оператора 2 видео по требованию или разработку и поддержание их

домашних страниц в интернете. Свое оборудование (например, видеосервер) такой оператор часто размещает в POP другого оператора, как это и показано на рисунке.

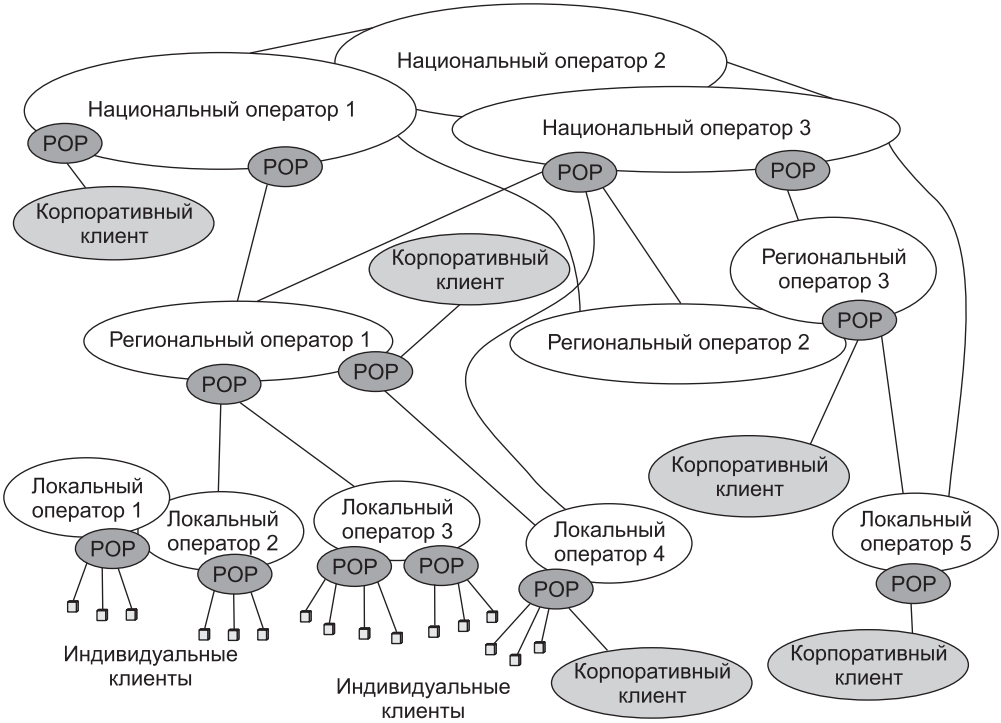


Рис. 18.2. Взаимоотношения между операторами связи различного типа

Организация интернета

Интернет представляет собой уникальную глобальную IP-сеть, так как почти все существующие компьютерные сети (за исключением разве что некоторых сетей, требующих особых мер защиты от вторжений и потому полностью изолированных от интернета) и отдельные компьютеры являются частью этой сети. Поэтому протокол IP является обязательным и единственным протоколом сетевого уровня, объединяющим все сети в интернете, уникальность которого проявляется во многих отношениях.

Прежде всего, интернет — это *самая большая в мире сеть*: по числу пользователей, по территории покрытия, по суммарному объему передаваемого трафика, по количеству входящих в ее состав сетей. Темпы роста интернета, хотя и снизились по сравнению с периодом интернет-революции середины 90-х, остаются очень высокими и намного превышают темпы роста телефонных сетей.

Интернет — это *сеть, не имеющая единого центра управления* и в то же время работающая по единым правилам и предоставляющая всем своим пользователям единый набор услуг. Интернет — это «сеть сетей», но каждая входящая в интернет сеть управляется независимым оператором — **провайдером услуг интернета** (Internet Service Provider, **ISP**).

Некоторые центральные органы существуют, но они отвечают только за единую ***техническую политику***, за согласованный набор технических стандартов, за централизованное назначение таких жизненно важных для гигантской составной сети параметров, как имена и адреса компьютеров и входящих в интернет сетей, но не за ежедневное поддержание сети в работоспособном состоянии. Такая высокая степень децентрализации имеет свои достоинства и недостатки.

Достоинства проявляются, например, в легкости наращивания интернета. Так, новому поставщику услуг достаточно заключить соглашение по крайней мере с одним из существующих провайдеров, после чего пользователи нового провайдера получают доступ ко всем ресурсам интернета. *Негативные* последствия децентрализации заключаются в сложности модернизации технологий и услуг интернета. Любые коренные изменения требуют согласованных усилий всех провайдеров услуг, тогда как в случае одного «собственника» они проходили бы намного легче. Недаром многие новые технологии пока применяются только в пределах сети одного поставщика. Примером может послужить технология групповой рассылки, которая очень нужна для эффективной организации аудио- и видеовещания через интернет, но все еще пока не может преодолеть границы, разделяющие сети различных провайдеров. Другой пример — не очень высокая надежность услуг интернета, так как никто из провайдеров не отвечает за конечный результат, например за доступ клиента *A* к сайту *B*, если они находятся в сетях разных поставщиков.

Стремительный рост числа пользователей интернета, привлекаемых информацией, содержащейся на его сайтах, изменил отношение корпоративных пользователей и операторов связи к этой сети. Сегодня интернет поддерживается практически всеми традиционными операторами связи. Кроме того, к ним присоединилось большое количество новых операторов, построивших свой бизнес исключительно на услугах интернета.

Поэтому общая структура интернета, показанная на рис. 18.3, во многом является отражением общей структуры всемирной телекоммуникационной сети, фрагмент которой мы уже видели на рис. 18.2.

Магистральные провайдеры услуг являются аналогами транснациональных операторов связи. Они обладают собственными транспортными магистралями, покрывающими крупные регионы (страна, континент, весь земной шар). Примерами магистральных провайдеров услуг являются такие компании, как **Verizon, AT&T, NTT, Tata Communications, Deutsche Telekom, Lumen Technologies**.

Соответственно, **региональные провайдеры услуг** оказывают услуги интернета в рамках определенного региона (область, графство, округ — в зависимости от принятого в той или иной стране административного деления), а **локальные провайдеры услуг** работают, как правило, в пределах одного города.

Связи между поставщиками услуг строятся на основе двухсторонних соглашений о взаимной передаче трафика. Такие соглашения называют **пиринговыми** (от англ. peer — равный в статусе). Магистральный оператор обычно имеет пиринговые соглашения со всеми остальными магистральными операторами (так как их немного), а региональные операторы, как правило, заключают такие соглашения с одним из магистральных операторов и с несколькими другими региональными операторами. Чтобы провайдерам было проще организовывать свои пиринговые связи, в интернете существуют специальные **центры обмена трафиком**, в которых соединяются сети большого количества провайдеров. Такие центры обмена обычно называются Internet eXchange Point (IXP) или Network Access Point (NAP).

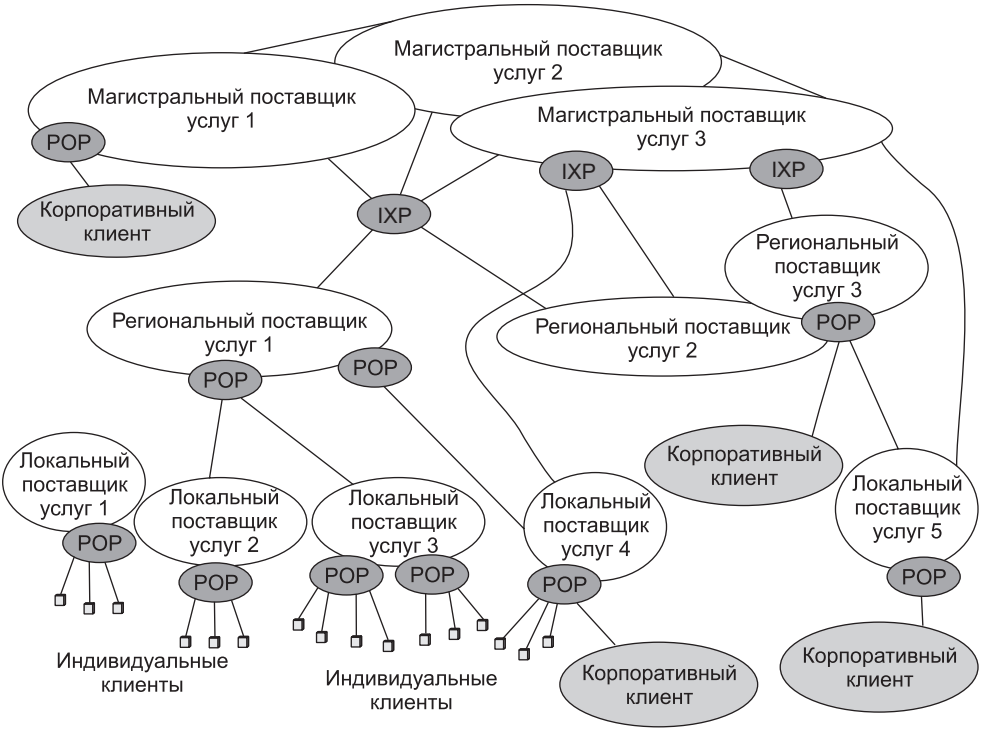


Рис. 18.3. Структура Интернета

Центр обмена трафиком является средством реализации пиринговых связей, предоставляя поставщикам услуг помещение и стойки для установки коммутационного оборудования. Все физические и логические соединения между своим оборудованием провайдеры услуг выполняют самостоятельно. Это значит, что не все сети провайдеров, которые пользуются услугами того или иного центра обмена данными, автоматически обмениваются трафиком друг с другом, а межсетевой обмен происходит только в том случае, когда между провайдерами заключено пиринговое соглашение и они его реализовали в данном центре обмена.

В интернете существует неофициальная градация *провайдеров интернета* по уровням (tiers) в зависимости от того, кто из них и кому платит за передачу транзитного трафика интернета. Провайдеры верхнего уровня (**Tier 1** — это, как правило, провайдеры интернационального и национального масштаба) могут достичь любой части интернета без платы за транзитный трафик — у них у всех имеются некоммерческие пиринговые соглашения друг с другом. Провайдеры второго уровня (**Tier 2**) относятся к смешанному типу — с одними провайдерами у них имеются некоммерческие пиринговые соглашения, с другими — договоры о плате за транзит своего трафика. Провайдеры третьего уровня (**Tier 3**) совсем не имеют бесплатных пиринговых соглашений и платят другим провайдерам за транзит своего трафика.

(S) *Многослойное представление технологий и услуг глобальных сетей*

(S) *Технологии доступа: PPP, HDLC, ADSL, PON, модемы*

ГЛАВА 19 Информационные услуги глобальных сетей

С точки зрения пользователей компьютерные сети представляют собой набор служб, предоставляющих разнообразные услуги: электронная почта, WWW, интернет-телефония, удаленный доступ к файлам, справочная служба, облачные вычисления и многие другие.

Заметим, что термин «service» в технической литературе переводится и как «сервис», и как «услуга», и как «служба». Хотя указанные термины иногда используются как синонимы, следует иметь в виду, что в некоторых случаях различие в значениях этих терминов носит принципиальный характер. Мы понимаем под «службой» сетевой программный компонент, который реализует предоставление некоторого набора услуг, а под «сервисом» — собственно тот набор услуг, который предоставляется службой. То есть термины «сервис» и «услуга» используются далее как синонимы. Все перечисленные службы предоставляют прикладные, они же информационные, сервисы и реализуются программным обеспечением, работающим на конечных узлах сети — компьютерах. Очевидно, что прикладные сервисы не могут предоставляться без обеспечивающих их транспортных сервисов, но эти функции сети скрыты от конечных пользователей.

В то же время характеристики транспортных сервисов, доставляющих информацию от одного конечного узла сети другому, существенно влияют на качество прикладных сервисов. Например, доставка пакетов интернет-телефонии с задержками более 100 мс сделает голос собеседника неузнаваемым, а предоставление видеосервису пропускной способности менее 5 Мбит/с приведет к многочисленным остановкам изображения высокого разрешения. Именно требования прикладных сетевых сервисов являются движущей силой всех изменений и усовершенствований в области транспортных технологий. Рост популярности мультимедийных сервисов реального времени (потоковое видео, аудио) привел к появлению новых протоколов транспортного уровня, переносящих временные отметки в отправленных пакетах, позволяющие приемной стороне контролировать темп поступления данных. Другим следствием растущей популярности мультимедийных сервисов стал взрывной рост объемов трафика, переносимого интернетом, что делает насущным дальнейший рост производительности транспортных сервисов этой глобальной сети.

В этой части книги вы познакомитесь с организацией наиболее популярных прикладных сетевых служб, предоставляющих свои услуги конечным пользователям: почтой, веб-службой и сетевым управлением на основе протокола SNMP.

Некоторые службы, а именно службы поддержки транспортных средств сети DNS и DHCP, выполняющие адресацию сетевых узлов, рассмотрены в предыдущих главах, учитывая их тесную связь с работой стека протоколов TCP/IP.

Существует и еще одна группа прикладных сетевых служб, которая обеспечивает безопасность сети, выполняя аутентификацию и авторизацию пользователей, шифрование трафика и ряд других важных операций, и которая рассматривается в последней части

книги, целиком посвященной средствам обеспечения безопасности. Там вы также найдете описание проблем безопасности, специфических для каждой из прикладных служб, ориентированных на конечного пользователя, то есть проблем безопасности веб-сервисов, почтовой службы и др.

Общие принципы организации сетевых служб

Службы принято делить на несколько групп по типам адресатов предоставляемых ими услуг:

- Службы, ориентированные на конечных пользователей и их приложений, такие как служба печати, файловый сервис, почта, веб-сервис, справочная служба, IP-телефония, служба облачных вычислений.
- Службы, обеспечивающие безопасность сети. К ним относятся сетевая аутентификация, авторизация и контроль доступа. Услуги этих служб требуются как конечным пользователям (например, при интерактивном входе в сеть), так и другим службам, которым необходимо защитить свою информацию и аутентифицировать своих пользователей (примером может быть служба баз данных, аутентифицирующая своих пользователей с помощью службы сетевой аутентификации).
- Службы, ориентированные на сетевых администраторов, решающих задачи конфигурирования и управления сетевыми устройствами; в эту категорию входят службы управления сетью на основе протоколов telnet и SNMP, служба мониторинга и аудита.
- Службы, помогающие компьютерам и сетевым устройствам предоставлять свои транспортные услуги: служба отображения символьных имен узлов на IP-адреса (DNS) и служба динамического назначения адресов (DHCP).
- Службы поддержки распределенных вычислений, например, служба репликации, служба вызова удаленных процедур (RPC), являющиеся вспомогательными по отношению к другим службам.

Клиентами сетевых служб могут быть другие сетевые службы, например, в число клиентов справочной службы входят служба аутентификации и почтовая служба.

В то же время служба, помимо основных услуг, дающих имя этому типу службы, может предоставлять и вспомогательные услуги. Например, веб-служба может выполнять аутентификацию самостоятельно, не обращаясь к централизованной службе сетевой аутентификации. Большая часть прикладных сетевых служб оформляются как приложения, то есть в виде исполняемых модулей стандартного для ОС, в среде которой они выполняются, формата. Поскольку такой же формат имеют и многие модули ОС, часто бывает сложно провести четкую грань между ОС и сетевыми службами. Решение о том, должна ли какая-то служба стать частью ОС или нет, принимает производитель последней.

В некоторых случаях для повышения производительности сервиса служба или ее определенные компоненты включаются в ядро. Примером являются клиентская и серверная части файловой службы, которые часто встраивают в ядро с тем, чтобы они могли получать быстрый прямой доступ ко всем модулям ОС без затрат времени на переключение режима из пользовательского в привилегированный.

Сетевые службы чаще всего представляют собой двухзвенные[1] *распределенные приложения*: одно из звеньев является клиентом, другое — сервером. Клиентская и серверная части в общем случае выполняются на разных компьютерах. Как правило, один сервер обслуживает большое число клиентов.

> Принципиальной разницей между клиентом и сервером является то, что инициатором выполнения сетевой службой некой работы всегда выступает клиент, а сервер всегда находится в режиме пассивного ожидания запросов. Например, почтовый сервер осуществляет доставку почты на компьютер пользователя только при поступлении запроса от почтового клиента.

В отличие от локальных приложений, которые, работая на одном компьютере, могут обмениваться данными через его оперативную память, части распределенного приложения, выполняемые на разных компьютерах, такой возможности не имеют. Взаимодействие клиента и сервера может выполняться только путем передачи *сообщений* через сеть в соответствии с выбранным *протоколом*.

Основными вопросами разработки распределенных приложений являются, во-первых, распределение функций между его звеньями (клиентом и сервером), а во-вторых, определение протокола взаимодействия этих звеньев.

Распределение функций между клиентом и сервером сетевой службы может выполняться различными способами. Например, клиент может быть наделен только функциями поддержки интерфейса с пользователем сервиса и поддержанием протокола взаимодействия, а вся логика работы службы возложена на серверную часть. Возможна и другая ситуация, когда клиент несет значительную нагрузку на поддержание работы сетевой службы. Например, при реализации почтовой службы на диске клиента может храниться локальная копия базы данных, содержащей его обширную переписку. В этом случае клиент делает основную работу при формировании сообщений в различных форматах, в том числе и сложном мультимедийном, поддерживает ведение адресной книги и выполняет много иных вспомогательных функций.

Протоколы обмена сообщениями, лежащие в основе сетевых служб, относятся к прикладному уровню. Службы, имеющие одно и то же назначение, могут использовать разные протоколы. К примеру, существуют сетевые файловые системы, построенные на основе принципиально отличающихся протоколов: FTP, SMB, NFS. Аналогично имеются два типа почтовой службы, в одной из них клиент и сервер взаимодействуют по протоколу SMTP, а в другой — X.400. Верно и обратное утверждение: службы, разработанные для предоставления разных сервисов, могут использовать один и тот же протокол взаимодействия клиентской и серверной частей. Например, протокол HTTP, разработанный для веб-службы, стал использоваться во многих службах и сетевых приложениях, например, в службе управления сетью, почтовой службе и многих других.

Пользовательский интерфейс, который в наше время обычно является графическим (Graphical User Interface, GUI), — важная часть клиента сетевой службы. От качества интерфейса зависит удобство работы пользователя с данной реализацией службы (степень дружественности). Он также отражает функциональное богатство службы. Заметим, различные реализации службы одного и того же назначения, поддерживающие один и тот же

---

[1] Распределенные приложения вообще и сетевые службы в частности могут иметь и многозвенную структуру.

прикладной протокол, могут значительно отличаться друг от друга функциональностью и дружественностью пользовательского интерфейса. Это хорошо видно на примере браузеров веб-службы — все они работают с одними и теми же веб-серверами и реализуют один и тот же протокол HTTP, но функциональность и дружественность браузера Chrome отличается от функциональности и дружественности браузера Microsoft Edge (хотя жесткая конкуренция между производителями браузеров заставляет их постоянно перенимать лучшие свойства продуктов конкурентов).

Веб-служба

Изобретение в 1989 году Тимом Бернерсом-Ли и Робертом Кайо **Всемирной паутины** (World Wide Web, **WWW**) стоит в одном ряду с изобретениями телефона, радио и телевидения. Благодаря этому изобретению интернет стал таким, каким мы его знаем сегодня. Используя Всемирную паутину, люди получили возможность доступа к нужной им информации в любое удобное для них время. Теперь проще найти интересующую вас статью в интернете, чем в стопке журналов, хранящихся рядом в шкафу. Очень быстро исчезают многие традиционные приемы рациональной организации работы с информацией, заключающиеся, например, в хранении полезной информации в записных книжках, раскладывании вырезок из журналов и газет в картонные папки с веревочками, упорядочивании документов в каталогах путем наклеивания на них маркеров с условными кодами, помогающими быстро отыскать нужный документ, и т. д. Этим приемам приходят на смену новые безбумажные технологии интернета, среди которых важнейшей является сетевая **служба WWW** (или, по-другому, **веб-служба**). Заметим, что веб-служба не только предоставляет любому человеку возможность быстрого поиска нужных данных и доступа к ним, но и позволяет ему выносить на многомиллионную аудиторию пользователей интернета собственную информацию — мнения, художественные и публицистические произведения, результаты научной работы, объявления и т. д. Причем он может это делать без особых организационных забот и практически бесплатно. Не будем долго останавливаться на описании всех возможностей этой службы, учитывая, что для большинства из нас регулярный просмотр веб-сайтов стал не просто обыденностью, а необходимым элементом жизненного уклада.

Веб- и HTML-страницы

Миллионы компьютеров, связанных через интернет, хранят невообразимо огромные объемы информации, представленной в виде веб-страниц.

Веб-страница, или **веб-документ**, как правило, состоит из основного HTML-файла и некоторого количества ссылок на другие объекты разного типа: JPEG- и GIF-изображения, другие HTML-файлы, аудио- и видеофайлы.

HTML-файлом, **HTML-страницей** или **гипертекстовой страницей** называют файл, который содержит текст, написанный на **языке HTML** (HyperText Markup Language — язык разметки гипертекста).

История появления языка HTML связана с попытками программистов разработать средство, позволяющее программным путем создавать красиво сверстанные страницы для просмотра на экране. Другими словами, красивая картинка появляется на дисплее

только в результате ее интерпретации специальной программой, а в исходном виде она представляет собой однообразный текст с множеством служебных пометок. Вместо применения различных приемов форматирования, таких как выделение заголовков крупным шрифтом, важных выводов — курсивным или полужирным начертанием и пр., создатель документа на языках этого типа просто вставляет в текст соответствующие указания о том, что данная часть текста должна быть выведена на экран в том или ином виде. Служебные пометки такого рода в исходном тексте выглядят, например, как `<b> </b>` (начать и закончить вывод текста полужирным начертанием) и называются **тегами**. Язык HTML не является первым языком разметки текста — его предшественники существовали задолго до появления веб-службы. Например, в первых версиях ОС Unix существовал язык troff (с помощью этого языка отформатированы страницы электронной документации Unix, известные как man-страницы).

В язык HTML включены разные типы тегов, команд и параметров, в том числе для вставки в текст изображений (тег `<img src=»…»`). Чтобы HTML-страница выглядела так, как задумал программист, она должна быть выведена на экран специальной программой, способной интерпретировать язык HTML. Такой программой является уже упоминавшийся веб-браузер[1].

Существует также особый тип тега, имеющий вид `<a href=»…»` `…</a>` и называемый **гиперссылкой**. Гиперссылка содержит информацию о веб-странице или объекте, который может находиться как на том же компьютере, так и на других компьютерах интернета. Отличие гиперссылки от других тегов состоит в том, что элемент, описываемый ею, не появляется автоматически на экране. Вместо этого на месте тега (гиперссылки) на экран выводится некоторое условное изображение или особым образом выделенный текст — имя гиперссылки. Чтобы получить доступ к объекту, на который указывает эта гиперссылка, пользователь должен щелкнуть на ней, дав тем самым команду браузеру найти и вывести на экран требуемую страницу или объект. После того как новая веб-страница будет загружена, пользователь сможет перейти по следующей гиперссылке — такой веб-серфинг может продолжаться теоретически сколь угодно долго. Все это время веб-браузер будет находить указанные в гиперссылках страницы, интерпретировать все размещенные на них указания и выводить информацию на экран в том виде, в котором ее спроектировали разработчики этих страниц.

URL-адрес

Браузер находит веб-страницы и отдельные объекты по адресам специального формата, называемым **URL** (Uniform Resource Locator — унифицированный указатель ресурса). URL-адрес может выглядеть, например, так: http://www.olifer.co.uk/books/books.htm. В URL-адресе можно выделить три части:

- *Тип протокола доступа* — в примере это http://.Помимо HTTP, здесь могут быть указаны и другие протоколы, такие как FTP, telnet, также позволяющие осуществлять удаленный доступ к файлам или компьютерам[2]. Тем не менее основным протоколом

---

[1] См. раздел «Сетевое программное обеспечение» главы 2.

[2] URL-адреса с самого начала предназначались не только для веб-служб, но и для других сервисов доступа к информации через интернет.

доступа к веб-страницам является HTTP (как в нашем примере), и мы поговорим о нем немного позже.

- *DNS-имя сервера* — это имя сервера, на котором хранится нужная страница, в примере www.olifer.co.uk.
- *Путь к объекту*. Обычно это составное имя файла (объекта) относительно главного каталога веб-сервера, предлагаемого по умолчанию. В нашем случае главным каталогом является /books/books.htm. По расширению файла мы можем сделать вывод о том, что это HTML-файл.

Веб-клиент и веб-сервер

Сетевая веб-служба представляет собой распределенную программу, построенную в архитектуре «клиент — сервер».

Клиент и сервер веб-службы взаимодействуют друг с другом по протоколу HTTP.

> Клиентская часть веб-службы, или **веб-клиент**, называемый также **браузером**, представляет собой приложение, которое устанавливается на компьютере конечного пользователя и предназначено для просмотра веб-страниц.

Одной из важных функций браузера является *поддержание графического пользовательского интерфейса*. Через интерфейс пользователь получает доступ к широкому набору услуг, главная из которых, конечно, веб-серфинг, включающий поиск и просмотр страниц, навигацию между уже просмотренными страницами, переход по закладкам и хранение истории посещений. Помимо средств просмотра и навигации, веб-браузер предоставляет пользователю возможность *манипулирования страницами*: сохранение их в файле на диске своего компьютера, вывод на печать, передача по электронной почте, контекстный поиск в пределах страницы, изменение кодировки и формата текста, а также множество других функций, связанных с представлением информации на экране и настройкой самого браузера.

К числу наиболее популярных сейчас браузеров можно отнести Microsoft Edge, Mozilla Firefox компании Mozilla, Google Chrome и Apple Safari. Но веб-браузер — это не единственный вид клиента, который может обращаться к веб-серверу. Эту роль могут исполнять любые программы и устройства, поддерживающие протокол HTTP. Значительную часть своих функций браузер выполняет в тесной кооперации с веб-сервером. Как уже отмечалось, клиент и сервер веб-службы связываются через сеть по протоколу HTTP. Это означает, что в клиентской части веб-службы присутствует клиентская часть протокола HTTP, а в серверной — серверная часть протокола HTTP.

> **Веб-сервер** — это программа, хранящая объекты локально в каталогах компьютера, на котором она запущена, и обеспечивающая доступ к этим объектам по URL-адресам.

Наиболее популярными веб-серверами сейчас являются Apache, Nginx и Microsoft Internet Information Server.

Как и любой другой сервер, веб-сервер должен быть постоянно в активном состоянии, прослушивая *TCP-порт 80*, являющийся назначенным портом протокола HTTP. При посту-

плении запроса от браузера устанавливается TCP-соединение. Браузер сообщает серверу имя файла, содержащего веб-страницу, например, в виде /books/abc.html, после чего сервер находит в своем каталоге этот файл и передает его браузеру. Браузер просматривает этот файл и для каждой ссылки на внешний объект — файл видеозаписи, файл изображения и т. п. — делает отдельный запрос к серверу. Получив объекты от сервера, веб-браузер отображает их на экране (рис. 19.1).

Передача объектов может происходить в рамках как одного TCP-соединения, так и нескольких — в зависимости от версии протокола HTTP.

В дополнительные функции сервера входят также аутентификация клиента и проверка прав доступа данного клиента к данной странице.

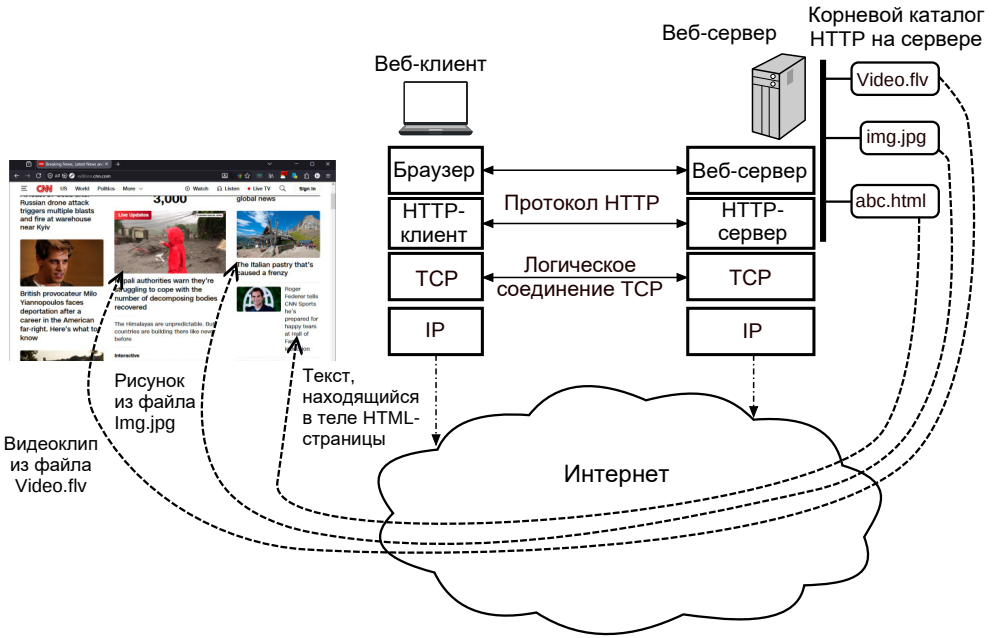


Рис. 19.1. Вывод веб-страницы на экран

Веб-сервер в отношении сеанса с веб-браузером является *сервером без сохранения состояния* (stateless). Это означает, что на сервере не хранится информация, касающаяся состояния сеанса: какие страницы пользователь уже посетил и какие данные ему были переданы. Такой режим общения с клиентом упрощает организацию сервера, которому необходимо отвечать на большой поток запросов различных пользователей, так что запоминание состояния сеансов пользователей существенно увеличило бы нагрузку на веб-сервер. Вместо этого веб-сервер рассматривает каждый запрос изолированно, отвечая на него и забывая про данного пользователя сразу после ответа. Кроме упрощения организации сервера, такой режим работы является более устойчивым, так как он не требует восстановления сеанса, если по той или иной причине он потерпел крах, оставляя пользователю заботы по решению этой проблемы. Недостатком данного режима является замедление работы клиента и увеличение трафика в сети из-за частого выполнения процедуры установления TCP-соединений.

Для повышения производительности некоторые веб-серверы прибегают к кэшированию наиболее часто используемых в последнее время страниц в своей памяти. Когда приходит запрос на какую-либо страницу, сервер, прежде чем считывать ее с диска, проверяет, не находится ли она в буферах более быстрой оперативной памяти. Кэширование страниц осуществляется и на стороне клиента, а также на промежуточных серверах (прокси-серверах). Кроме того, эффективность обмена данными с клиентом иногда повышают путем компрессии (сжатия) передаваемых страниц. Объем передаваемой информации уменьшают также за счет того, что клиенту передается не весь документ, а только та часть, которая была изменена. Все эти приемы повышения производительности веб-службы реализуются средствами протокола HTTP.

Протокол HTTP

Протокол передачи гипертекста **HTTP** (HyperText Transfer Protocol) — это протокол прикладного уровня, регламентирующий обмен информацией (HTML-страницами, изображениями и др.) между веб-сервером и веб-клиентом. Этот протокол работает поверх транспортного протокола TCP (по умолчанию номер порта — 80).

Существует несколько версий (рис. 19.2) протокола: HTTP 1.0, HTTP 1.1, HTTP/2 и HTTP/3.

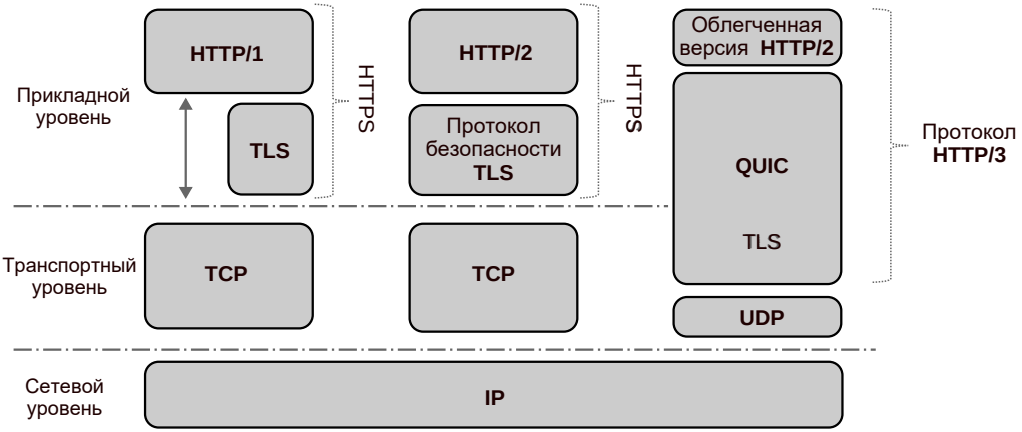


Рис. 19.2. Стеки протоколов HTTP

Обмен сообщениями идет по обычной схеме «запрос-ответ». Клиент и сервер обмениваются ***текстовыми*** сообщениями стандартного формата, то есть каждое сообщение представляет собой несколько строк обычного текста в кодировке ASCII. Для транспортировки HTTP-сообщений служит протокол TCP.

Версии протокола HTTP различаются способом транспортировки данных от сервера к браузеру через сеть. Каждый последующий вариант, сохраняя в целом синтаксис и семантику сообщений, устранял недостатки механизма передачи запросов и ответов между веб-браузером и веб-сервером, и в то же время привносил новые проблемы. Рассмотрим эти изменения последовательно от версии к версии.

Версия **HTTP/1.0**. *Установление одного TCP-соединения для передачи одного запроса и ответа на него.*

В версии HTTP 1.0 поддерживается только режим кратковременных соединений, когда после передачи одного запроса и получения ответа, то есть передачи только одного объекта веб-страницы, TCP-соединение закрывается. Такой режим полностью соответствует концепции сервера без сохранения состояния, а это, как уже отмечалось, приводит к замедлению работы браузера и увеличению трафика из-за частого выполнения процедуры трехэтапного установления TCP-соединения.

Версия **HTTP/1.1**. *Установление одного TCP-соединения для передачи последовательности пар «запрос-ответ».*

В каждый момент времени в рамках одного TCP-соединения **HTTP/1.1** обрабатывает только один запрос. Для обработки следующего запроса может устанавливаться новое TCP-соединение, но может использоваться и уже существующее. В этом случае оно называется *долговременным* (keep-alive) и по нему последовательно передаются несколько пар «запрос-ответ». Долговременное соединение, в свою очередь, может быть использовано двумя способами:

- *Последовательная передача запросов с простоями* — это способ, посредством которого новый запрос посылается только после получения ответа от сервера.
- *Конвейерная передача* — это более эффективный способ, в котором следующий запрос посылается до прибытия ответа на один или несколько предыдущих запросов (напоминает метод скользящего окна). Обычно по умолчанию степень параллелизма устанавливается на уровне 5–10 запросов, но у пользователя имеется возможность изменять этот параметр при конфигурировании клиента. Большинство браузеров не используют конвейеризацию, поскольку она вызывает проблемы на практике.

Этот способ передачи имеет серьезный дефект — **блокировку начала очереди** (head-of-line, HoL). Поскольку ответы должны отправляться в том же порядке, в котором были отправлены запросы, медленный ответ на очередной запрос может задержать все последующие ответы по тому же соединению. Например, объемный объект, состоящий из многих байтов, такой, например, как видеоклип, может выкачиваться долго и тем самым вызвать «затор» — блокировать следующие за ним запросы на другие более мелкие объекты (рисунки, тексты, заголовки). Из-за этого пользователь браузера может испытывать регулярные раздражающие паузы.

Для того чтобы повысить скорость выполнения запросов к веб-серверам и для избежания блокировок производители браузеров при использовании протокола HTTP/1.1 стали поддерживать открытие нескольких (обычно до шести) TCP-соединений с одним и тем же сервером для параллельной загрузки ресурсов. Несколько соединений используются для параллельного извлечения нескольких различных объектов одной и той же страницы (объекты передаются только целиком). Поскольку как большие, так и маленькие объекты отображаются на экране браузера параллельно, передача объемного видеоклипа не приводит к «замораживанию» экрана дисплея.

Однако такой подход приводит к другим проблемам. Как известно, установление каждого TCP-соединения требует времени, установление же нескольких параллельных TCP-соединений существенно сказывается на времени обмена данными между сервером и клиентом. Увеличение количества TCP-соединений приводит также к увеличению количества сокетов, что для веб-сервера, обслуживающего множество клиентов, может

привести к исчерпанию ресурсов. Есть и еще один недостаток. Поскольку протокол TCP обеспечивает примерно равное разделение общей пропускной способности канала между всеми TCP-соединениями, проходящими через этот канал, то имеется потенциальная возможность злоупотреблений, связанная с «захватом» пропускной способности клиентами или серверами HTTP, путем распараллеливания передачи данных. Именно с наличием всех этих недостатков связано то, что возможность открытия параллельных TCP-соединений не поддерживается стандартом HTTP/1.1.

Версия **HTTP/2**. *Установление одного TCP-соединения, поддерживающего мультиплексирование потоков.*

Прежде всего давайте еще раз определим, что понимается под термином «поток» в данном контексте. **Поток** (stream) — это независимая последовательность байтов, которая переносит одну пару «запрос-ответ», а иногда и несколько последовательных пар «запрос-ответ». Поток может быть однонаправленным или двунаправленным, может создаваться и закрываться независимо от других. Например, в HTTP/1.1 через *одно* TCP-соединение проходит *один* поток.

В HTTP/2 TCP-соединение передает сразу *несколько* потоков, каждый из которых несет одну пару «запрос-ответ». На рис. 19.3 показана схема преобразования данных, передаваемых между браузером и веб-сервером. Несколько запросов, сгенерированных браузером, образуют потоки A, B и C, которые обрабатываются протоколом HTTP/2. Эти несколько потоков **мультиплексируются**, то есть каждый поток разделяется на кадры (фреймы), которые чередуются и в «перемешанном» виде передаются на уровень TCP для передачи через одно TCP-соединение. Работа по разделению HTTP-сообщений на части (кадрирование) и формирование из этих частей общего мультиплексированного потока выполняется протоколом HTTP.

Когда данные приходят на другую сторону соединения, они передаются в неизменном виде протоколу HTTP, который разделяет мультиплексированный (смешанный) поток на исходные подпотоки. Восстановленные сообщения передаются для обработки обычным образом либо веб-серверу, либо веб-браузеру. По дороге данные шифруются/дешифрируются протоколом TLS.

Версия HTTP/2 ускорила процесс построения веб-браузером веб-страницы за счет следующих усовершенствований:

- Возможность разбивать HTTP-сообщение на независимые кадры, чередовать их и затем собирать их заново на другом конце — самое важное улучшение HTTP/2. Вместо использования отдельных TCP-соединений для передачи каждого запроса и ответа, приводившего к простоям из-за того, что новый запрос не может быть послан без получения ответа, теперь используется одно TCP-соединение для мультиплексирования нескольких запросов, которые могут быть посланы практически одновременно.
- Введена приоритизация запросов к веб-серверу, благодаря которой сервер знает, какой запрос более важен веб-браузеру для построения страницы.
- Поддерживается режим *Server Push*, при котором веб-сервер может передать веб-браузеру не только запрашиваемые ресурсы, но и те, которые, по мнению веб-сервера, скоро понадобятся веб-браузеру.
- Используется компрессия заголовков сообщений HTTP, значительно сокращающая длину сообщения за счет сжатия таких потенциально длинных полей, как куки.

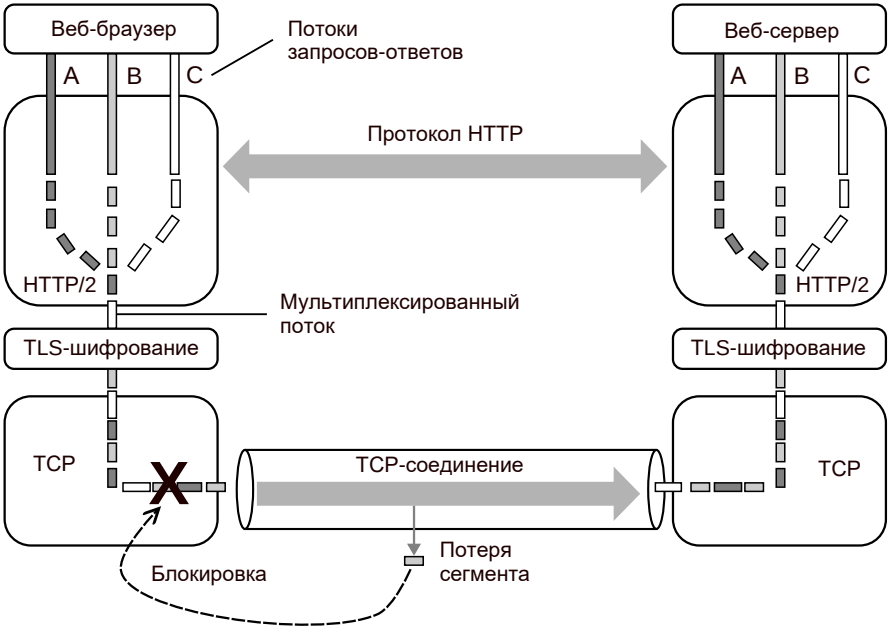


Рис. 19.3. Схема мультиплексирования потоков в протоколе HTTP/2

Однако снова возникла та же проблема — блокировка начала очереди, на этот раз из-за *потери* сегментов TCP. Как подробно описано в главе 15, TCP гарантирует упорядоченную доставку байтов. Если сегмент TCP потерян, процесс передачи данных блокируется на время, необходимое для повторной передачи потерянных данных. Важно заметить, что протокол TCP ничего не знает о потоках, составляющих мультиплексированный поток. Потерянный сегмент TCP представляется ему безличной последовательностью байтов, никак не связанной ни с одним потоком. TCP блокирует весь агрегированный поток, то есть потеря одного сегмента останавливает *все потоки*, использующие это TCP-соединение, до тех пор, пока потерянный сегмент не будет восстановлен. Особенно сильно этот недостаток HTTP/2 проявляется в беспроводных сетях с частыми потерями пакетов (Wi-Fi, сотовых). Реализованный в следующей версии, HTTP/3, новый механизм мультиплексирования потоков устранил этот недостаток.

Прежде чем перейти к рассмотрению HTTP/3, коротко остановимся на протоколе TLS. По умолчанию трафик HTTP между веб-сервером и браузером не шифруется, следовательно, есть большой риск, что злоумышленник может прочитать или модифицировать данные. Для снижения рисков используется протокол безопасности TLS (Transport Layer Security)[1], решающий следующие задачи:

- аутентификация сервера, основанная на сертификатах;
- шифрование данных;
- обеспечение целостности данных.

[1] Подробнее о безопасности и протоколе TLS читайте в разделе «Технологии защищенного канала» главы 31.

Протокол TLS, так же как и протокол HTTP, работает на прикладном уровне. Его услугами помимо HTTP могут пользоваться почтовые протоколы IMAP и SMTP, протокол передачи данных FTP и многие другие.

Версии HTTP/1.0 и HTTP/1.1 могут работать как с протоколом TLS, так и без него. HTTP/2, как правило, требует TLS. Названия безопасных версий протоколов, работающих поверх TLS, дополняются буквой S, как показано на рис. 19.2 — **HTTPS**.

Протокол HTTP/3 и транспортный протокол QUIC

Протокол **HTTP/3** (RFC 9114) представляет собой комбинацию *облегченной версии HTTP/2 и протокола QUIC*. HTTP/3 всегда работает в защищенном режиме, поскольку протокол TLS интегрирован в протокол QUIC (рис. 19.4).

В то время как HTTP/1.1 и HTTP/2 используют для транспортировки сообщений протокол TCP, HTTP/3 работает исключительно через протокол QUIC.

> Протокол **QUIC** (Quick UDP Internet Connection) — сравнительно новый протокол прикладного уровня, предназначенный для повышения производительности транспортных сервисов безопасного HTTP, обеспечивающий надежную передачу данных с контролем перегрузки между двумя конечными точками на основе логического соединения между веб-браузером и веб-сервером. Функциональность протокола QUIC сопоставима с объединенной функциональностью протоколов TCP, TLS и частично HTTP/2.

По мнению разработчиков стандарта, протокол QUIC обладает рядом функций, желательных для HTTP-транспорта, таких как:

- установление соединения с низкой задержкой;
- мультиплексирование потоков;
- управление потоком (flow control) на уровне отдельных потоков (streams)[1].

Рассмотрим последовательно, как эти декларируемые свойства реализованы в протоколе QUIC.

Чтобы обеспечить *низкую задержку установления соединения*, было предпринято несколько принципиальных усовершенствований. Во-первых, протокол QUIC разрабатывался с нуля как *специализированный* (а не универсальный) транспортный протокол для HTTP, оптимизированный для учета конкретных требований веб-приложений. Во-вторых, в качестве базового транспортного протокола QUIC использует не медленный, но надежный протокол TCP, а быстрый, хотя и менее надежный дейтаграммный протокол UDP. На его основе QUIC предоставляет протоколу HTTP собственный более быстрый сервис логических соединений. В-третьих, благодаря тому что протокол безопасности TLS встроен в протокол QUIC, шифрование выполняется по умолчанию, без необходимости обращения к отдельному уровню TLS. Кроме того, совмещается установление соединений, предназначенных для обеспечения надежности передачи данных (как в TCP), с установлением защищенных каналов протокола безопасности TLS, поэтому общее время установления соединения между клиентом и сервером HTTP/3 оказывается меньше, чем время, затрачиваемое HTTP/2 на установление соединений двух видов — TCP и TLS.

---

[1] Очень важно различать здесь и далее использование терминов «flow» и «stream», которые оба переводятся на русский язык как «поток».

Мультиплексирование потоков реализуется протоколом QUIC более эффективно, чем в HTTP/2. Как показано на схеме (рис. 19.3), задачи формирования мультиплексированного потока и передачи мультиплексированного потока решаются отдельно протоколами HHTP/2 и TCP соответственно. Поэтому протокол TCP ничего не знает о потоках, а следовательно, не может избирательно, попоточно управлять трафиком: если нужно повторить передачу потерянного сегмента, то блокируется не отдельный поток, а все данные, передаваемые в этот момент через логическое соединение TCP.

Совсем другую картину мы наблюдаем на рис. 19.4: здесь протокол QUIC отвечает и за формирование мультиплексированного потока, и за установление логического соединения, и за передачу по этому соединению мультиплексированного потока. Поэтому протокол QUIC обладает информацией о том, как связаны суммарный поток, передаваемый через QUIC-соединение, и составляющие его потоки (на рисунке потоки *A*, *B* и *C*), а именно: данные каких потоков содержатся в том или ином фрагменте суммарного потока. Благодаря этому потеря QUIC-соединением пакета блокирует только тот поток, чьи данные потеряны, а остальные потоки продолжают передаваться. (На рисунке: из-за потери пакета, содержащего данные потока *A*, блокируется только поток *A*.)

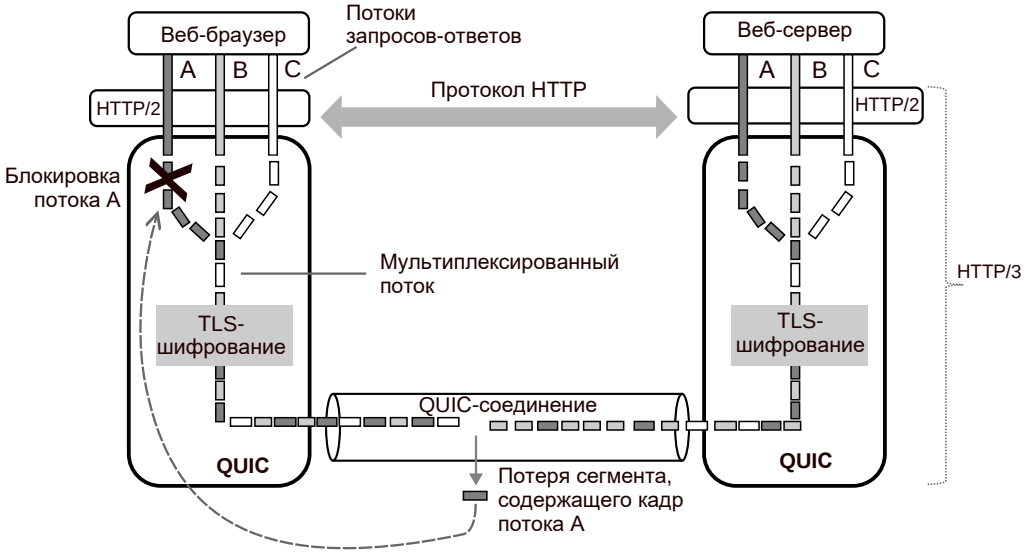


Рис. 19.4. Схема мультиплексирования потоков в протоколе HTTP/3

Запросы и ответы генерируются веб-приложением и в виде потока байтов направляются протоколу HTTP, а затем протоколу QUIC. Для каждого потока стек QUIC создает буфер отправки (в буфере хранятся сообщения приложения, состоящие из заголовков и данных, в виде байтов, которые еще не отправлены или не подтверждены), *смещение* (используется в случае потери данных для нахождения дубликата и их повторной передачи) и некоторые другие переменные.

Все потоки независимы друг от друга и имеют уникальные идентификаторы.

На стороне отправителя QUIC разделяет буфер отправки каждого потока на отдельные фрагменты/**кадры**. Далее планировщик протокола QUIC решает, какие кадры отправлять

следующими. Он просматривает все потоки, учитывая готовность данных, приоритеты, наличие блокировок, а также признак того, что данные требуют повторной передачи. Принимается во внимание также состояние окон управления потоком и перегрузками (об этом сказано ниже). В результате формируется упорядоченный список кадров-кандидатов на отправку или, другими словами, *мультиплексированный поток*.

Затем QUIC шифрует кадры и создает UDP-дейтаграммы, которые передаются в рамках логического QUIC-соединения на приемную сторону, где протокол QUIC расшифровывает, обрабатывает кадры, отправляет подтверждения/обновления управления потоком. В случае потери данных отправитель повторно передает недостающие байты потока (как новые кадры с теми же смещениями).

Таким образом, передача сразу нескольких потоков внутри одного QUIC-соединения позволяет нескольким запросам и ответам выполняться практически одновременно, не блокируя друг друга.

Управление потоком (flow control) в протоколе QUIC так же, как и в протоколе TCP, состоит в согласовании скоростей работы отправителя и получателя таким образом, чтобы соблюсти баланс: с одной стороны, буфер получателя не переполнялся, а с другой — отправитель не простаивал слишком долго. Для решения такой задачи протокол использует квитанции, таймеры и окна управления потоком, которые определяют, с какой интенсивностью отправитель должен отсылать в сеть пакеты, когда он должен замедлить или приостановить отправку, сколько времени он должен ждать подтверждения для отправленных данных, сколько байтов он может отправить, не получив квитанции на предыдущие, и т. д.

Протокол QUIC управляет потоком не только на уровне QUIC-соединения (connection), но и на уровне отдельных потоков (streams). Помимо **окна соединения** (connection_window), протокол QUIC поддерживает отдельные **окна потока** (stream_window) для каждого потока. Окно потока ограничивает объем данных потока, который отправитель может передать без получения квитанций, и тем самым гарантирует, что буфер получателя не переполнится данными быстрее, чем веб-приложение успеет прочитать их. При открытии нового потока обе стороны договариваются о начальном размере окна. Получатель по мере обработки и чтения веб-приложением данных из своего буфера периодически отправляет отправителю специальные кадры, которые сообщают ему, сколько места в буфере освободилось. Получив такой кадр, отправитель изменяет свое доступное окно потока и может отправить больше или меньше данных.

Окна потоков работают в тандеме с общим окном соединения: отправитель может отправить данные, только если ему это позволяют ограничения как окна потока, так и окна соединения. Окно потока предотвращает переполнение буфера конкретного потока, в то время как окно соединения защищает общие ресурсы получателя и помогает в управлении глобальной скоростью передачи данных.

Таким образом, механизм окна потока обеспечивает избирательный контроль над каждым отдельным потоком, гарантируя, что даже если некоторый поток очень активен, он не использует все ресурсы получателя и не блокирует обработку других потоков.

Встроенные механизмы мультиплексирования и установления соединений в протоколе QUIC позволяют HTTP/3 также более эффективно решать задачу управления *перегрузкой*, чем это делают HTTP/2 с протоколом TCP. В числе прочих усовершенствований отметим следующие:

- Благодаря включению в квитанцию времени между ее отправкой и получением подтверждаемого пакета, в QUIC более точно рассчитывается время оборота RTT, а это позволяет отличать кратковременные задержки от реальной перегрузки.
- В отличие от TCP, который уменьшает **окно перегрузки** (congestion window) при возникновении тайм-аута, протокол QUIC делает это, только если диагностируется «устойчивая перегрузка» (persistent congestion). Это предотвращает неоправданное замедление работы сети из-за единичных потерь данных, особенно это важно в беспроводных сетях, в которых потери пакетов часто вызываются помехами, а не перегрузкой.
- Возможность простой модификации механизма борьбы с перегрузками программистом. Поскольку код протокола QUIC располагается не в ядре, как у протокола TCP, а в пользовательском пространстве, то внесение изменений в программу управления перегрузкой существенно упрощается, что позволяет настраивать алгоритм для конкретных приложений и для конкретного состояния сети.

Формат HTTP-сообщений

В протоколе HTTP все сообщения состоят из текстовых строк. HTTP-сообщения бывают двух типов — запросы и ответы, — имеющих единую обобщенную структуру, состоящую из трех частей:

- стартовая строка;
- заголовки;
- тело сообщения.

Стартовая строка является обязательной, а заголовки и тело сообщения — опциональными.

HTTP-запрос	**HTTP-ответ**
`GET /books/books.html HTTP/1.1`	`HTTP/1.1 200 OK`
`Host: www.olifer.co.uk`	`Date: 1 Jan 2019 14:00:30`
`User-agent: Mozilla/5.0`	`Server: Apache/1.3.0 (Unix)`
`Accept-language: ru`	`Content-Length: 1234`
`Connection: close`	`Connection: close`
Тело сообщения	Тело сообщения

С помощью этого запроса браузер хочет получить от веб-сервера www.olifer.co.uk с помощью протокола HTTP/1.1 веб-страницу с именем books.html, находящуюся в каталоге /books. При этом браузер сообщает свою версию (Mozilla/5.0) и язык страницы — ru (Russian).

В ответе сервер сообщает, что он смог выполнить запрос (код 200 OK) и отправил ответ размером 1234 байта 1 января 2019 года в 14:00. Сервер также сообщает версию своего программного кода — Apache/1.3.0(Unix).

В табл. 19.1 приведены форматы стартовых строк и заголовков для помещенных выше примеров запроса и ответа.

Как видно из таблицы, запросы и ответы имеют разные форматы стартовой строки. Каждая из них состоит из трех элементов, включающих поле *версии протокола HTTP*. И в запросе, и в ответе примера указана версия HTTP 1.1.

Таблица 19.1. Форматы стартовых строк и заголовков

Обобщенная структура сообщения	HTTP-запрос	HTTP-ответ
Стартовая строка (всегда должна быть первой строкой сообщения; обязательный элемент)	Формат запроса Метод/ URL HTTP/1.x. Пример: GET /books/books.htm HTTP/1.1	Формат ответа: HTTP/1.x КодСостояния Фраза. Пример: HTTP/1.1 200 OK
Заголовки (следуют в произвольном порядке; могут отсутствовать)	Заголовок о DNS-имени компьютера, на котором расположен веб-сервер. Пример: Host: www.olifer.co.uk	Заголовок о времени отправления данного ответа. Пример: Date: 1 Jan 2009 14:00:30
	Заголовок об используемом браузере. Пример: User-agent: Mozilla/5.0	Заголовок об используемом веб-сервере. Пример: Server: Apache/1.3.0 (Unix)
	Заголовок о предпочтительном языке. Пример: Accept-language: ru	Заголовок о количестве байтов в теле сообщения. Пример: Content-Length: 1234
	Заголовок о режиме соединения. Пример: Connection: close	Заголовок о режиме соединения. Пример: Connection: close
Пустая строка		
Тело сообщения (может отсутствовать)	Здесь могут быть расположены ключевые слова для поисковой машины или страницы для передачи на сервер	Здесь может быть расположен текст запрашиваемой страницы

Стартовая строка запроса включает в себя поле *метода* — это название операции, которая должна быть выполнена. Чаще всего в запросах используется метод GET, то есть запрос объекта (именно он включен в наш пример запроса). Еще одним элементом стартовой строки является URL-адрес запрашиваемого объекта — здесь это имя файла /books/books.htm.

Помимо метода GET в запросах протокол предусматривает и другие методы, такие как HEAD, POST, PUT, DELETE и др.

- Метод HEAD аналогичен методу GET, но запрашиваются только метаданные заголовка HTML-страницы.
- Метод POST используется клиентом для отправки данных на сервер: сообщений электронной почты, ключевых слов в запросе поиска, веб-формы.
- Метод PUT используется клиентом для размещения некоторого объекта на сервере, на который указывает URL-адрес.
- Метод DELETE указывает серверу на то, что некоторый объект на сервере, определяемый URL-адресом, необходимо удалить.

Методы GET и HEAD считаются безопасными[1] для сервера, так как они только передают информацию клиенту, а методы POST, PUT и DELETE — опасными, поскольку передают информацию на сервер. Наибольшую угрозу представляют два последних метода, так как они непосредственно указывают на объект на сервере. Используя эти методы, злоумышленник может атаковать сервер, заменяя или удаляя некоторые его объекты.

[1] Вопросы безопасности веб-службы подробно обсуждаются в главе 32.

В стартовой строке ответа, помимо уже упоминавшегося указания на версию протокола HTTP, имеется поле *кода состояния* и поле *фразы* для короткого текстового сообщения, поясняющего данный код пользователю. В настоящее время стандарты определяют пять классов кодов состояния:

- 1*xx* — информация о процессе передачи;
- 2*xx* — информация об успешном принятии и обработке запроса клиента (в таблице в примере стартовой строки ответа приведен код и соответствующая фраза `200 OK`, сообщающий клиенту, что его запрос успешно обработан);
- 3*xx* — информация о том, что для успешного выполнения операции нужно произвести следующий запрос по другому URL-адресу, указанному в дополнительном заголовке `Location`;
- 4*xx* — информация об ошибках со стороны клиента (читатель наверняка не раз сталкивался с ситуацией, когда при указании адреса несуществующей страницы браузер выводил на экран сообщение `404 Not Found`);
- 5*xx* — информация о неуспешном выполнении операции по вине сервера (например, сообщение `505 http Version Not Supported` говорит о том, что сервер не поддерживает версию HTTP, предложенную клиентом).

Среди кодов состояния имеется код 401, сопровождаемый сообщением `authorization required`. Если клиент получает такое сообщение в ответ на попытку доступа к странице или объекту, это означает, что доступ к данному ресурсу ограничен и требует авторизации пользователя. Помимо поясняющей фразы сервер помещает в свой ответ дополнительный заголовок `www-Authenticate:<... >`, который сообщает клиенту, какую информацию он должен направить серверу для того, чтобы процедура авторизации могла быть выполнена. Обычно это имя и пароль. Веб-клиент с момента получения такого ответа сервера начинает добавлять во все свои запросы к ресурсам данного сервера дополнительный заголовок `Authorization`: *<имя, пароль>*, который содержит информацию, необходимую для авторизации доступа.

Динамические и статические веб-страницы

До сих пор мы подразумевали, что содержание страницы не изменяется в зависимости от действий пользователя. Когда пользователь щелкает на гиперссылке, он переходит на *новую* страницу, а если выполняет команду возвращения обратно, то на экране снова появляется предыдущая страница в *неизменном* виде. Такие страницы называются **статическими**.

Однако в некоторых случаях желательно, чтобы содержание страницы изменялось в зависимости от действий пользователя, например, при наведении указателя мыши на определенную область страницы там появлялся рисунок вместо текста или значка. Динамическое воспроизведение состояния базы данных также является типичным примером ситуации, когда статическая страница не может решить задачу. Так, многие интернет-магазины поддерживают базу данных продаваемых товаров, а вывод количества оставшихся в наличии товаров требует динамического обновления соответствующего поля веб-страницы.

Веб-страницы, которые могут генерировать выводимое на экран содержание, меняющееся в зависимости от некоторых внешних условий, называются **динамическими**. Динамика страницы достигается путем ее программирования, обычно с использованием программных

языков сценариев: Perl, PHP и JavaScript. Различают два класса программ, предназначенных для создания динамического содержания веб-страниц:

- программы, работающие на стороне клиента (то есть на том компьютере, где запущен веб-браузер, воспроизводящий страницу на экране);
- программы, работающие на стороне сервера.

Если программа работает на стороне клиента, то код страницы передается веб-сервером веб-браузеру как обычный статический объект, а затем браузер выполняет этот код, с его помощью создает динамическое содержание страницы и выводит ее на экран. Существуют различные механизмы создания динамических страниц на стороне клиента — это прежде всего надстройки браузера, Java-апплеты, JavaScript-сценарии и ActiveX-элементы.

Механизм *надстроек браузера* (add-on) позволяет расширить его функциональные возможности за счет динамического вызова из браузера дополнительных программ, установленных на клиентском компьютере. Программа-надстройка обрабатывает объекты веб-страницы определенного типа, например, программа-надстройка Adobe Acrobat NPAPI Plugin вызывается браузером Firefox для показа пользователю документа в формате PDF в окне браузера, а программа-надстройка Shockwave Flash вызывается для проигрывания видеоклипов в формате Flash или для интерактивной анимации, написанной на языке ActionScript. Нужно отметить, что термин «надстройка» считается обобщенным названием *разных* видов надстроек браузера — например, браузера Firefox, который различает несколько видов надстроек:

- *расширения* (extensions) встраиваются в браузер (то есть становятся его частью);
- *темы* изменяют внешний вид окна браузера и также встраиваются в него;
- *вставки* (plug-ins) — это программы, оформленные чаще всего в виде библиотек и вызываемые через стандартный для браузера интерфейс, такой как, например, NPAPI (Netscape Plugin Application Programming Interface). Вставки являются внешними по отношению к браузеру программами.

Java-апплеты представляют собой скомпилированные программы, которые написаны на языке Java, динамически загружаются браузером с веб-сервера и выполняются виртуальной Java-машиной (JVM) клиентского компьютера. Передача Java-апплета Java-машине выполняется вставкой Java Applet Plugin.

Динамическое содержание страницы может также создаваться с помощью JavaScript-сценариев — специальных программ (скриптов) на языке JavaScript, разработанном компанией Netscape. JavaScript, хотя и имеет общую часть в названии с языком Java, является самостоятельным языком со своим синтаксисом. В отличие от Java — языка компилирующего типа — JavaScript является языком интерпретирующего типа, интерпретатором которого выступает браузер.

Динамическое содержание страницы может быть также создано предложенными компанией Microsoft управляющими ActiveX-элементами, которые могут вызывать внешние объекты и встраивать результаты их работы в страницу. ActiveX-элементы являются двоичными исполняемыми файлами, которые должны иметь цифровую подпись. Ограничением ActiveX-элементов является то, что их выполнение возможно только в среде ОС Windows на процессоре Intel x86 или же при их эмуляции. На практике это означает, что страницы с ActiveX-элементами правильно воспроизводятся только браузером Internet Explorer.

При программировании содержания страницы на стороне сервера процесс выглядит немного сложнее, так как программный код страницы создает содержание (контент) на сервере, следовательно, здесь нужен дополнительный этап — передача этого содержания по протоколу HTTP на клиентскую машину браузеру. Популярными языками сценариев для серверной части являются Perl, ASP, JSP и PHP. Существует также стандартный программный интерфейс между веб-сервером и программами, генерирующими динамическое содержание, — это общий шлюзовой интерфейс (Common Gateway Interface, CGI).

Многие из перечисленных выше средств расширения функциональных возможностей браузера являются слабыми звеньями в обеспечении безопасности клиентского компьютера, на котором они работают (см. главу 32). Более безопасно использование средств создания динамического содержания за счет функций языка HTML5, в котором имеются специальные теги для проигрывания аудио- и видеоклипов.

Почтовая служба

Сетевая почтовая служба, или **электронная почта**, — это распределенное приложение, главной функцией которого является предоставление пользователям сети возможности обмениваться электронными сообщениями.

Как и все сетевые службы, электронная почта построена в архитектуре «клиент — сервер». Почтовый клиент всегда располагается на компьютере пользователя, а почтовый сервер, как правило, работает на выделенном компьютере.

Почтовый клиент (называемый также **агентом пользователя**) — это программа, предназначенная для поддержания пользовательского интерфейса (обычно графического), а также для предоставления пользователю широкого набора услуг по подготовке электронных сообщений. В число таких услуг входит создание текста в различных форматах и кодировках, сохранение, уничтожение, переадресация, сортировка писем по разным критериям, просмотр перечня поступивших и отправленных писем, грамматическая и синтаксическая проверка текста сообщений, ведение адресных баз данных, автоответы, образование групп рассылки и прочее, и прочее. Кроме того, почтовый клиент поддерживает взаимодействие с серверной частью почтовой службы.

Почтовый сервер выполняет прием сообщений от клиентов, для чего постоянно находится в активном состоянии. Кроме того, он выполняет буферизацию сообщений, распределение поступивших сообщений по индивидуальным буферам (почтовым ящикам) клиентов, управляет объемами памяти, выделяемой клиентам, выполняет регистрацию клиентов и регламентирует их права доступа к сообщениям, а также решает много других задач.

Электронные сообщения

Почтовая служба оперирует **электронными сообщениями** — информационными структурами определенного стандартного формата. Упрощенно электронное сообщение может быть представлено в виде двух частей, одна из которых (заголовок) содержит вспомогательную информацию для почтовой службы, а другая (тело сообщения) — это, собственно, то «письмо», которое предназначается для прочтения, прослушивания или просмотра адресатом (RFC 822).

Главными элементами заголовка являются адреса отправителя и получателя в виде Polina@domen.com, где Polina — идентификатор пользователя почтовой службы, а domen.com — имя домена, к которому относится этот пользователь. Кроме этого, почтовая служба включает в заголовок дату и тему письма, делает отметки о применении шифрования, срочности доставки, необходимости подтверждения факта прочтения этого сообщения адресатом и др. Дополнительная информация заголовка может оповещать почтового клиента получателя об использовании той или иной кодировки. При транспортировке через интернет почтовое сообщение помещается в **конверт** (envelope), который также имеет несколько служебных полей, например поле отправителя и поля получателей (рис. 19.5). Информация конверта используется только при транспортировке почтового сообщения, а информация заголовка сообщения — почтовым клиентом получателя.

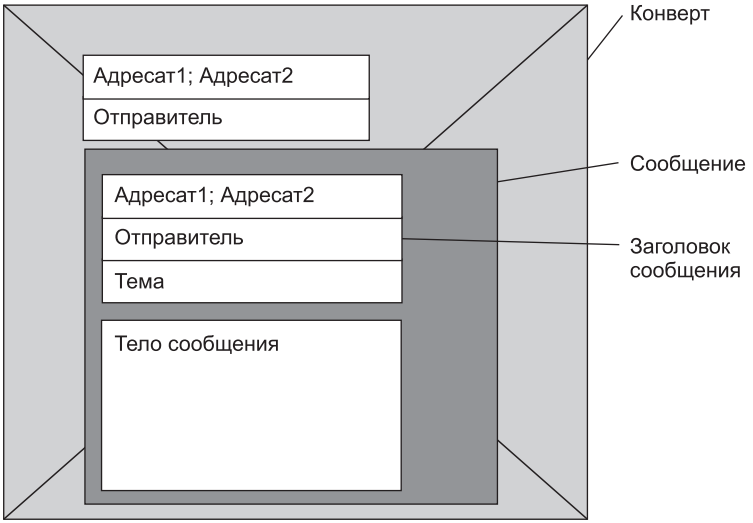


Рис. 19.5. Конверт и сообщение электронной почты Интернета

Первоначально тело сообщения представляло собой сплошной текст в кодировке ASCII (такая же кодировка использовалась для служебных полей конверта и заголовка сообщения).

Большая популярность электронной почты привела к ее интернационализации, что заставило принять несколько новых стандартов, разрешающих применять в теле сообщения не только ASCII-коды, но и такие коды, как UTF-8, позволяющие пользователям всех стран задействовать свой родной язык при написании электронного письма. Ограничение на использование ASCII осталось только для полей конверта, и это ограничение приходится преодолевать несколько искусственным способом, преобразуя исходные символы, не относящиеся к кодировке ASCII, в более длинную последовательность ASCII-кодов (например, используя популярный в системах электронной почты алгоритм base64).

Важную роль в расширении возможности электронной почты по передаче мультимедийной информации сыграл стандарт **MIME** (Multipurpose Internet Mail Extensions — многоцелевые расширения почты интернета). Этот стандарт описывает структуру сообщения, состо-

ящего из нескольких частей, каждая из которых имеет свои заголовок и тело. Заголовок описывает тип данных, которые содержатся в теле; это могут быть как обычные текстовые данные в формате ASCII, так и данные другого типа, например:

- текст в 8-битном формате (такая возможность описана в документе RFC 6152, принятом в марте 2011 года);
- текст не в формате ASCII, преобразованный в ASCII-код (например, с помощью уже упомянутого алгоритма base64);
- гипертекст (HTML);
- изображение;
- видеоклип;
- звуковой файл.

Части отделяются друг от друга последовательностью символов, называемой *границей* (boundary); граница не должна встречаться в теле частей сообщения.

В заголовке каждой части сообщения имеется также информация о том, каким образом почтовый клиент должен обрабатывать тело части — отображать ее немедленно при открытии сообщения (например, встраивая изображение в текст) или считать это тело *вложением* (attachment), которое пользователь будет обрабатывать сам. Одна из спецификаций стандарта MIME (каждая спецификация MIME описывает одно или несколько расширений оригинальной спецификации RFC 822), а именно — RFC 1847, относится к расширениям безопасности, поэтому этот документ называют спецификацией **S/MIME** (Security MIME). В S/MIME описаны два новых типа частей MIME:

- цифровая подпись (Mutipart/Signed);
- шифрованное тело (Multipart/Encrypted).

Эти два типа частей сообщения могут использоваться вместе с целью обеспечения аутентичности, целостности и конфиденциальности электронного письма.

Протокол SMTP

В качестве средств передачи сообщения почтовая служба интернета использует стандартный, разработанный специально для почтовых систем протокол **SMTP** (Simple Mail Transfer Protocol — простой протокол передачи почты). Этот протокол является одним из первых стандартизованных протоколов прикладного уровня, дата его публикации — август 1982 года. Как и большинство других протоколов прикладного уровня, SMTP реализуется несимметричными взаимодействующими частями: SMTP-клиентом, работающим на стороне отправителя, и SMTP-сервером, работающим на стороне получателя. SMTP-сервер должен постоянно быть в режиме подключения, ожидая запросов со стороны SMTP-клиента. Логика протокола SMTP действительно является достаточно простой, как это и следует из его названия:

1. После того как, применяя графический интерфейс своего почтового клиента, пользователь щелкает на значке отправки сообщения, SMTP-клиент посылает запрос на установление TCP-соединения на порт 25 SMTP-сервера (это назначенный порт).
2. Если сервер готов, то он посылает свои идентификационные данные, в частности свое DNS-имя. Если SMTP-сервер оказался не готов, то он посылает соответствующее со-

общение клиенту, и тот снова посылает запрос, пытаясь заново установить соединение. Затем клиент передает серверу почтовые адреса (имена) отправителя и получателя. Если имя получателя соответствует ожидаемому, то после получения адресов сервер дает согласие на установление SMTP-соединения, и в рамках этого логического канала происходит передача сообщения.

3. Если после приема тела сообщения сервер отвечает командой OK, то это означает, что сервер принял на себя ответственность по дальнейшей передаче сообщения получателю. Однако это не означает, что сервер гарантирует успешную доставку, потому что последнее зависит не только от него: например, клиентская машина получателя может быть в течение длительного времени не подсоединена к интернету. Если сервер не может доставить сообщение, то он передает отчет об ошибке отправителю сообщения и разрывает соединение.

Используя одно TCP-соединение, клиент может передать несколько сообщений, предваряя каждое из них указанием почтовых адресов отправителя и получателя. После завершения передачи сообщения TCP- и SMTP-соединения разрываются, и благополучно переданное сообщение сохраняется в буфере на сервере.

Нужно отметить, что в протоколе SMTP предусмотрены как положительные, так и отрицательные уведомления о доставке (промежуточной или окончательной) электронного письма. Однако только отрицательные уведомления являются обязательными, поэтому обычно SMTP-серверы предпочитают не передавать положительные уведомления в направлении отправителя.

ПРИМЕЧАНИЕ

Хотя в любом протоколе предполагается обмен данными между взаимодействующими частями, то есть данные передаются в обе стороны, различают протоколы, ориентированные на передачу (push protocols), и протоколы, ориентированные на прием данных (pull protocols). В протоколах, ориентированных на передачу, к которым, в частности, относится протокол SMTP, клиент является инициатором передачи данных на сервер, а в протоколах, ориентированных на прием, к которым относятся, например, протоколы HTTP, POP3 и IMAP, клиент является инициатором получения данных от сервера.

Непосредственное взаимодействие клиента и сервера

Обсудив основные составляющие почтовой службы, рассмотрим несколько основных схем ее организации. Начнем с простейшего, практически не используемого сейчас варианта, когда отправитель непосредственно взаимодействует с получателем. Как показано на рис. 19.6, у каждого пользователя на компьютере установлены почтовые клиент и сервер.

Данила, используя графический интерфейс своего почтового клиента, вызывает функцию создания сообщения, в результате чего на экране появляется стандартная незаполненная форма сообщения, в поля которой Данила вписывает свой адрес, адрес Полины и тему письма, а затем набирает текст письма. При этом он может пользоваться не только встроенным в почтовую программу текстовым редактором, но и привлекать для этой цели другие программы, например MS Word. Когда письмо готово, Данила вызывает функцию отправки сообщения, и встроенный SMTP-клиент посылает запрос на установление связи SMTP-серверу на компьютере Полины. В результате устанавливаются SMTP- и TCP-соединения,

после чего сообщение передается через сеть. Почтовый сервер Полины сохраняет письмо в памяти ее компьютера, а почтовый клиент по команде Полины выводит его на экран, при необходимости выполняя преобразование формата. Полина может сохранить, переадресовать или удалить это письмо. Если Полина решит направить электронное сообщение Даниле, то схема работы почтовой службы будет симметричной.

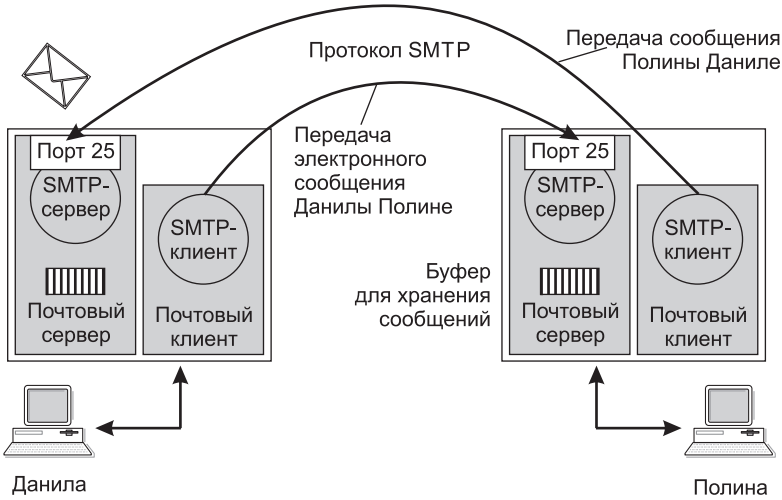


Рис. 19.6. Схема непосредственного взаимодействия клиента и сервера

Схема с выделенным почтовым сервером

Рассмотренная только что простейшая схема почтовой связи кажется работоспособной, однако у нее есть серьезный и очевидный дефект. Мы упоминали, что для обмена сообщениями необходимо, чтобы SMTP-сервер постоянно находился в ожидании запроса от SMTP-клиента, то есть для того, чтобы письма, направленные Полине, доходили до нее, ее компьютер должен постоянно находиться в режиме подключения. Понятно, что такое требование для большинства пользователей неприемлемо. Естественным решением этой проблемы является размещение SMTP-сервера на специально выделенном для этой цели компьютере-посреднике. Это должен быть достаточно мощный и надежный компьютер, способный круглосуточно передавать почтовые сообщения от многих отправителей ко многим получателям. Обычно почтовые серверы поддерживаются крупными организациями для своих сотрудников или провайдерами для своих клиентов. Для каждого домена имен система DNS создает записи типа MX, хранящие DNS-имена почтовых серверов, обслуживающих пользователей, относящихся к этому домену. На рис. 19.7 представлена схема с выделенным почтовым сервером. Чтобы не усложнять рисунок, мы показали на нем только компоненты, участвующие в передаче сообщения от Данилы к Полине. Для обратного случая схема должна быть симметрично дополнена.

1. Итак, пусть Данила решает послать письмо Полине, для чего он запускает на своем компьютере установленную на нем программу почтового клиента (например, Microsoft Outlook или Mozilla Thunderbird). Он пишет текст сообщения, указывает необходимую сопроводительную информацию, в частности адрес получателя polina@bim.com,

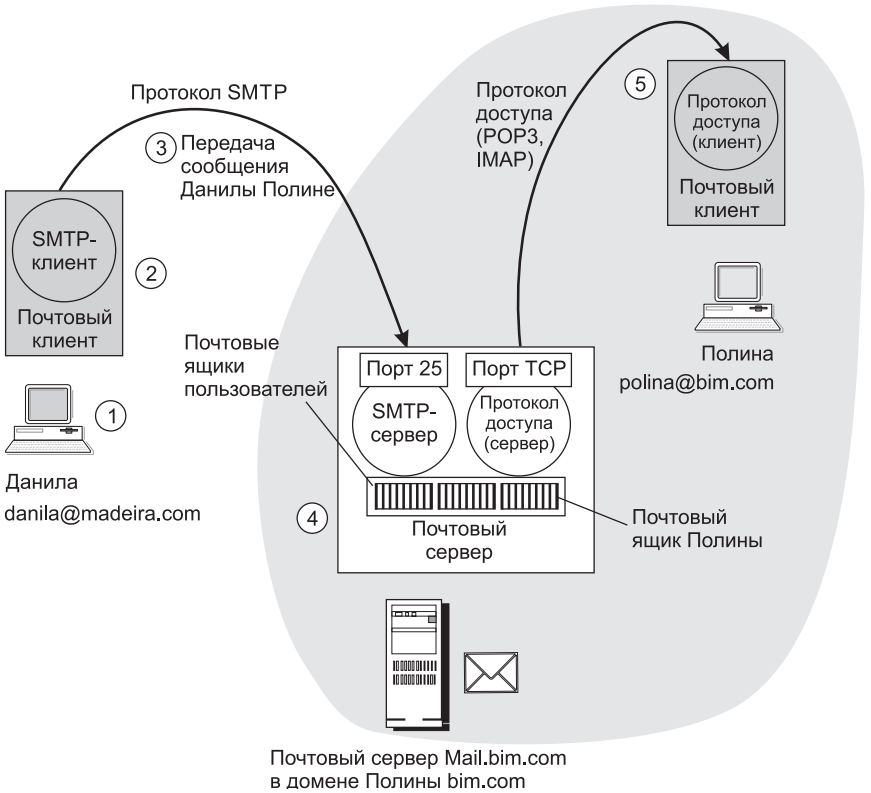


Рис. 19.7. Схема с выделенным почтовым сервером в принимающем домене

и щелкает мышью на значке отправки сообщения. Поскольку готовое сообщение должно быть направлено совершенно определенному почтовому серверу, клиент обращается к системе DNS, чтобы определить имя почтового сервера, обслуживающего домен Полины bim.com. Получив от DNS в качестве ответа имя mail.bim.com, SMTP-клиент еще раз обращается к DNS — на этот раз чтобы узнать IP-адрес почтового сервера mail.bim.com.

2. SMTP-клиент посылает по данному IP-адресу запрос на установление TCP-соединения через порт 25 (SMTP-сервер).
3. С этого момента начинается диалог между клиентом и сервером по протоколу SMTP, с которым мы уже знакомы. Заметим, что здесь, как и у всех протоколов, ориентированных на передачу, направление передачи запроса от клиента на установление SMTP-соединения совпадает с направлением передачи сообщения. Если сервер оказывается готовым, то после установления TCP-соединения сообщение Данилы передается.
4. Письмо сохраняется в буфере почтового сервера, а затем направляется в индивидуальный буфер, отведенный системой для хранения корреспонденции Полины. Такого рода буферы называют почтовыми ящиками. Важно заметить, что, помимо Полины, у почтового сервера имеется и много других клиентов, что усложняет его работу. Таким

образом, почтовый сервер должен решать самые разнообразные задачи по организации многопользовательского доступа, включая управление разделяемыми ресурсами и обеспечение безопасного доступа.

5. В какой-то момент, который принципиально не связан с моментом поступления сообщений на почтовый сервер, Полина запускает свою почтовую программу и выполняет команду проверки почты. После этой команды почтовый клиент должен запустить протокол доступа к почтовому серверу. Однако это не будет SMTP. Напомним, что протокол SMTP используется тогда, когда необходимо передать данные на сервер, а Полине, напротив, нужно получить их с сервера.

Для этого случая были разработаны другие протоколы, обобщенно называемые протоколами доступа к почтовому серверу, такие, например, как **POP3** и **IMAP**. Оба этих протокола относятся к протоколам, ориентированным на прием данных. Инициатором передачи сообщений от почтового сервера почтовому клиенту по протоколу POP3 или IMAP является клиент. Почтовый сервер ожидает запрос на установление TCP-соединения по протоколу POP3 через порт 110, а по протоколу IMAP — через порт 143 (на рисунке эти порты обобщенно изображены как порт TCP). В результате работы любого из них письмо Данилы оказывается в памяти компьютера Полины. Заметим, что на этот раз направление запроса от клиента к серверу не совпадает с направлением передачи данных, показанному стрелкой.

Оба протокола — POP3 и IMAP — поддерживают передачу отправителю квитанций, подтверждающих доставку, а также факт открытия сообщения получателем в случае, когда отправитель запрашивает такую почтовую услугу. Обычно для обеспечения приватности почтовый клиент не отправляет квитанцию автоматически, а спрашивает у получателя разрешение на такое действие.

Схема с двумя почтовыми серверами-посредниками

Прежде чем мы перейдем к сравнению двух протоколов доступа к почте, давайте рассмотрим еще одну схему организации почтовой службы, наиболее приближенную к реальности (рис. 19.8). Здесь передача сообщений между клиентами почты (на рисунке между отправителем Данилой и получателем Полиной) проходит через два промежуточных почтовых сервера, каждый из которых обслуживает домен своего клиента. На каждом из этих серверов установлены и клиентские части протокола SMTP. При отправке письма почтовый клиент Данилы передает сообщение по протоколу SMTP почтовому серверу домена, к которому относится Данила, — RoyalMail.madeira.com. Это сообщение буферизуется на данном сервере, а затем по протоколу SMTP передается дальше на почтовый сервер домена Полины — mail.bim.com, откуда описанным уже образом попадает на компьютер Полины.

Возникает вопрос: зачем нужна такая двухступенчатая передача через два почтовых сервера? Прежде всего для повышения надежности и гибкости процедуры доставки сообщения. Действительно, в схеме с передачей сообщения сразу на сервер получателя почтовый клиент отправителя в случае неисправности почтового сервера должен самостоятельно справляться со сложившейся нештатной ситуацией. Если же посредником в передаче сообщения является другой почтовый сервер, то это позволяет реализовывать разнообразные логические механизмы реакции на отказы на стороне сервера, который к тому же всегда находится в режиме подключения. Например, при невозможности

передать письмо почтовому серверу получателя сервер отправляющей стороны может не только рапортовать об этом своему клиенту, но и предпринимать собственные действия — пытаться снова и снова послать письмо, повторяя эти попытки в течение достаточно длительного периода.

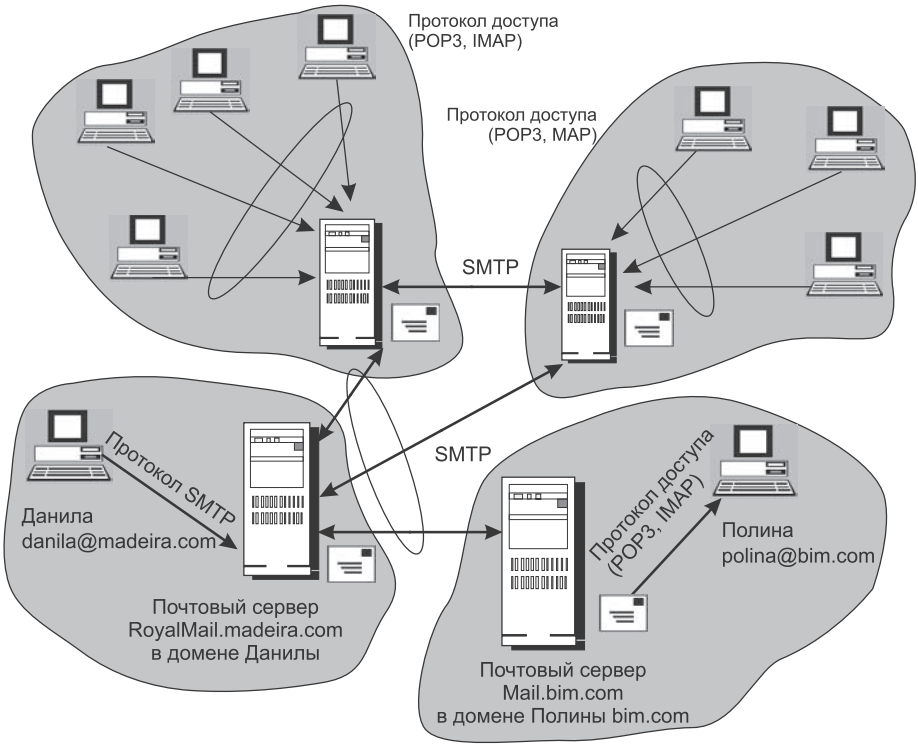


Рис. 19.8. Схема с выделенными почтовыми серверами в каждом домене

Протоколы POP3 и IMAP

Теперь сравним два протокола доступа к почте: **POP3** (Post Office Protocol v.3 — протокол почтового отделения версии 3) и **IMAP** (Internet Mail Access Protocol — протокол доступа к электронной почте интернета). Оба протокола решают одну и ту же задачу — обеспечивают пользователей доступом к их корреспонденции, хранящейся на почтовом сервере. В связи с многопользовательским характером работы почтового сервера оба протокола поддерживают аутентификацию пользователей на основе идентификаторов и паролей пользователей. Однако протоколы POP3 и IMAP имеют и принципиальные различия, важнейшее из которых состоит в следующем. Получая доступ к почтовому серверу по протоколу POP3, вы перекачиваете адресованные вам сообщения в память своего компьютера, при этом на сервере не остается никакого следа от считанной вами почты. Если же доступ осуществляется по протоколу IMAP, то в память вашего компьютера передаются только копии сообщений, хранящихся на почтовом сервере.

Это различие серьезно влияет на характер работы с электронной почтой. Сейчас очень распространенной является ситуация, когда человек в течение одного и того же периода времени использует несколько разных компьютеров: на постоянном месте работы, дома, в командировке. Теперь давайте представим, что произойдет с корреспонденцией пользователя Полины, если она получает доступ к почте по протоколу POP3. Письма, прочитанные на работе, останутся в памяти ее рабочего компьютера. Придя домой, она уже не сможет прочитать их снова. Опросив почту дома, она получит все сообщения, которые поступили с момента последнего обращения к почтовому серверу, но из памяти сервера они исчезнут, и завтра на работе она, возможно, не обнаружит важные служебные сообщения, которые были загружены на диск ее домашнего ноутбука. Таким образом, получаемая Полиной корреспонденция «рассеивается» по всем компьютерам, которыми она пользовалась. Такой подход не позволяет рационально организовать почту: распределять письма по нескольким различным папкам, сортировать их по разным критериям, отслеживать состояние переписки, отмечать письма, на которые послан ответ, и письма, еще требующие ответа, и т. д. Конечно, если пользователь всегда работает только с одним компьютером, недостатки протокола POP3 не столь критичны. Но и в этом случае проявляется еще один недостаток протокола — клиент не может пропустить, не читая, ни одного письма, поступающего от сервера. То есть объемное и, возможно, совсем не нужное вам сообщение может надолго заблокировать вашу почту.

Протокол IMAP был разработан как ответ на эти проблемы. Предположим, что теперь Полина получает почту по протоколу IMAP. С какого компьютера она бы ни обратилась к почтовому серверу, ей будут переданы только копии запрошенных сообщений. Вся совокупность полученной корреспонденции останется в полной сохранности в памяти почтового сервера (если, конечно, не поступит специальной команды от пользователя об удалении того или иного письма). Такая схема доступа делает возможным для сервера предоставление широкого перечня услуг по рациональному ведению корреспонденции, то есть именно того, чего лишен пользователь при применении протокола POP3. Важным преимуществом IMAP является также возможность предварительного чтения заголовка письма, после чего пользователь может принять решение о том, есть ли смысл получать с почтового сервера само письмо.

Облачные сервисы

Концепция облачных вычислений

До недавнего времени пользователи компьютерной сети твердо знали, на каких компьютерах работают программы, обрабатывающие их данные. Это были либо их собственные компьютеры, на которых пользователи запускали текстовые процессоры или программы подготовки презентаций, либо корпоративный сервер, размещенный в центре данных предприятия.

Новая концепция организации вычислений, получившая название **облачных вычислений** (cloud computing), изменяет привычный мир пользователя, так как в соответствии с ней компьютер, который выполняет программу пользователя, находится где-то в облаке вычислительных ресурсов — процессоров, оперативной памяти, дисковых накопителей. Облачные вычисления позволяют разгрузить пользовательский компьютер и перене-

сти вычисления на некоторые удаленные компьютеры, связанные с пользовательским компьютером через сеть. Это облако может принадлежать как коммерческому провайдеру, так и самому предприятию, но его организация скрыта от пользователя. При этом пользователь не заботится о том, на каком именно компьютере он должен запустить свою программу, достаточно ли у компьютера ресурсов для качественного выполнения программы в данный момент, достаточна ли пропускная способность сети для быстрого получения реакции программы на своем мониторе, — он просто запускает программу, а затем работает с ней, как если бы работал с ней локально. Результаты также можно хранить в облаке, это надежнее, так как провайдер облачных вычислений заботится об их сохранности.

ПРИМЕЧАНИЕ

Сама по себе интерпретация сети как облака не нова. Эту метафору применяли всегда, подчеркивая тот факт, что внутренняя организация сети не важна для конечных пользователей — для них важно, что сеть может передавать данные между двумя компьютерами. Такое облако можно назвать «транспортным». Облачные сервисы имеют дело с вычислительным облаком, то есть таким, которое ориентировано на вычисления.

Облачные вычисления являются сравнительно новым видом услуг, но по прогнозам специалистов они станут значительной вехой в процессе эволюции компьютерных сетей. Можно сказать, что с появлением этих услуг эволюция компьютерных сетей завершила виток, вернувшись к первоначальной модели, когда сеть соединяла терминал пользователя с мейнфреймом, на котором пользователь запускал свою программу. Это действительно так, но, как и всегда, виток эволюции привел к появлению нового качества, так как теперь «тупые» терминалы превратились в мощные компьютеры, перепоручающие облачным серверам выполнение только некоторых, возможно, критичных программ, оставляя возможность работать с остальными программами локально.

Хорошую аналогию, поясняющую значение облачных вычислений для развития общества, дал Николас Карр в своей книге The Big Switch[1]. Он сравнил появление облачных вычислений в информационном веке с электрификацией в индустриальном веке. До электрификации страны каждая организация, каждый производитель и каждый домовладелец должны были сами заботиться о выработке энергии — для этого существовали водяные колеса, ветряные мельницы, паровые машины. С распространением электрификации исчезла необходимость вырабатывать энергию самостоятельно — стало достаточным подключиться к электрической сети. Карр считает, что облачные вычисления открывают новую эру в использовании компьютеров. Сейчас организации обеспечивают себя компьютерными ресурсами самостоятельно. В будущем предполагается, что организация будет просто подключаться к облаку и получать вычислительные ресурсы, которые ей нужны. И если нужно быстро увеличить вычислительную мощность или объем хранимых данных, то облако легко предоставит дополнительные ресурсы своему клиенту точно так же, как электрическая сеть легко справляется с дополнительной нагрузкой, когда вы включаете микроволновую печь или электрическую дрель в дополнение к работающему телевизору и настольной лампе (рис. 19.9).

[1] http://www.amazon.co.uk/The-Big-Switch-Rewiring-Edison/dp/0393333949.

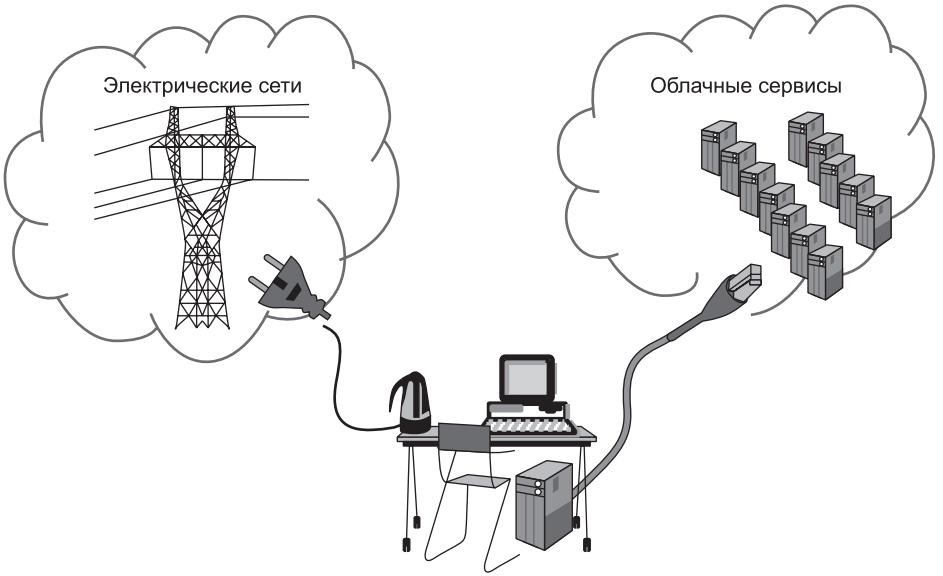


Рис. 19.9. Стратегическое значение облачных вычислений

Определение облачных вычислений

Существуют различные определения облачных вычислений. Приведем определение организации NIST, активно участвующей в разработке этого нового вида услуг:

> Облачные вычисления - это модель, предоставляющая удобный доступ по требованию к разделяемому пулу конфигурируемых вычислительных ресурсов, которые могут быть быстро выделены пользователю и отданы обратно в пул с минимальными затратами на управление этим процессом или с минимальным взаимодействием с провайдером услуг[1].

С этим определением связаны следующие свойства облачных вычислений:

- *Качественно новый уровень разделения ресурсов.* В отличие от прежних моделей вычислений, в которых вычислительные ресурсы принадлежали одному владельцу (ресурсы корпоративной сети, хотя и разделяются между пользователями, принадлежат одной и той же организации-владельцу), облачные вычисления основаны на модели, где вычислительные ресурсы разделяются между многими арендаторами-клиентами и владельцем-провайдером на всех уровнях — уровне сети, хоста и приложений.
- *Высокая степень масштабируемости.* Хотя отдельная организация может владеть сотнями или даже несколькими тысячами серверов, облачная среда позволяет масштабировать вычислительную мощность до десятков тысяч серверов, в разы наращивая пропускную способность каналов доступа и объем хранилища данных. Для обеспечения

[1] The NIST Definition of Cloud Computing, Special Publication 800-145, NIST, August 2011.

такой масштабируемости провайдер создает большое количество центров данных, распределенных географически.

- *Эластичность.* Пользователи могут быстро наращивать и уменьшать вычислительные ресурсы по мере необходимости. Таким свойством обычная корпоративная информационная система не обладает.
- *Гибкость оплаты использованных ресурсов.* Пользователи платят только за те ресурсы, которые они действительно задействовали, и за тот период времени, в течение которого эти ресурсы использовались.
- *Самостоятельное выделение ресурсов.* Пользователи могут управлять выделением необходимых им ресурсов самостоятельно через удобный интерфейс, предоставляемый провайдером. Скорее всего, эти операции будут выполнять сотрудники ИТ-отдела предприятия, а не рядовые пользователи.

Облачные вычисления появились как результат развития технологий, применяющихся в коммерческих центрах данных при оказании разнообразных услуг хостинга. В конце концов сетевое сообщество осознало, что концепция использования компьютерной сети стала меняться — вместо гибкого соединения жестко заданных пользователем узлов (как правило, клиентского компьютера и сервера) теперь происходит гибкое соединение клиентского компьютера с гибко настраиваемым пулом вычислительных ресурсов. При этом пул ресурсов может быть рассредоточен по различным центрам данных, находящимся в разных городах и, возможно, странах. Важно понимать, что облачные вычисления — не столько новая технология, сколько *комбинация* уже существовавших ранее технологий. Эти технологии-компоненты развивались различными темпами и в разных условиях, причем изначально они не создавались для работы как единое целое. Но их смогли «притереть» друг к другу и создать новое качество — облачные среды. Новые достижения в области процессоров, технологий виртуализации, системах дисковой памяти, широкополосном доступе в интернет, быстрые и недорогие серверы внесли свой вклад в превращение облачных вычислений в очень привлекательную модель.

Основополагающей технологической платформой, на которой базируются облачные вычисления, является **виртуализация**. Термин «виртуализация» относится к абстрагированию компьютерных ресурсов (процессора, памяти, дисковой памяти, сети, стека протоколов и баз данных) от прикладных программ и конечных пользователей облачного сервиса. Технологии виртуализации позволяют многочисленным арендаторам облака видеть его ресурсы как выделенные только для них. Виртуализация применяется и в центрах данных, принадлежащих предприятию, — она улучшает использование ресурсов и упрощает операционную эффективность ИТ-отдела. Технологии виртуализации применимы к ресурсам разного вида: компьютеру и ОС, хранилищам данных и базам данных, приложениям и сетям (VLAN, MPLS VPN 2-го и 3-го уровня и др.).

Операторы облачных сервисов применяют различные технологии виртуализации и программирования сетей своих центров данных, в том числе технологии SDN и NFV, рассмотренные в главе 16. Существует большое количество программных систем, помогающих оператору облачных сервисов автоматизировать процесс создания услуг для его многочисленных пользователей. Одной из таких систем является OpenStack — свободно распространяемая программная среда с открытым исходным кодом для виртуализации физических ресурсов трех типов: вычислительных, хранения данных и сетевых.

Модели облачных сервисов

По способу реализации облачные среды делятся на публичные, частные и гибридные.

- *Публичные облака* создаются провайдерами услуг, и их сервисы предоставляются предприятиям или частным лицам как публичным клиентам.
- *Частные облака* создаются некоторыми предприятиями для использования только сотрудниками этих предприятий, то есть как часть корпоративной сети предприятий.
- *Гибридные облака* — это комбинация публичных и частных облаков, которыми пользуется некое предприятие.

Далее рассматриваются три основные модели сервисов, реализуемых в публичных облачных средах и предоставляемых провайдерами облачных вычислений.

1. **Приложения как сервис (S**oftware- **as a S**ervice, **SaaS)**. Традиционный метод покупки программного обеспечения заключается в загрузке программы в собственный компьютер после оплаты лицензии на применение этой программы (капитальные затраты). Пользователь может купить и контракт на обслуживание, чтобы получать «заплатки» и обновления программы. Таким образом, пользователь сам заботится о совместимости программы с ОС, установке обновлений и удовлетворении условий лицензии.

   В модели SaaS пользователь не покупает программное обеспечение, а *арендует* его для работы на условиях подписки или оплаты за использование. То есть вместо капитальных затрат пользователь несет текущие (операционные) расходы. В некоторых случаях при ограниченном применении сервиса (например, программа выполняется не более часа в сутки) сервис может быть бесплатным.

   Нужно подчеркнуть, что модель приложения как сервиса отличается от услуг хостинга приложений в двух существенных аспектах. Во-первых, в модели SaaS приложение используется в режиме разделения времени всеми арендаторами (то есть различными организациями), которые подписались на доступ к данному приложению, в то время как приложение хостинга выделяется в единоличное пользование организации-клиенту (хотя и разделяется между сотрудниками этой организации). Во-вторых, программы для хостинга приложений часто пишутся без учета работы через сеть, в то время как программы модели SaaS всегда оптимизированы для сетевого доступа. Примером SaaS-сервисов являются сервисы Google Apps, Microsoft office 365, Apple iWork.

2. **Платформа как сервис (P**latform **as a S**ervice, **PaaS)**. В этой модели провайдер предоставляет среду для разработчиков программного обеспечения. Как правило, в нее входят набор средств разработчика (SDK) и стандарты разработки программ, каналы распространения программ и механизмы оплаты. Этот сервис направлен на поддержку быстрой разработки приложений с низким уровнем начальных вложений и применением устоявшихся каналов для быстрого нахождения пользователей новых программ. Пример этого типа облачных сервисов — Microsoft Azure.

3. **Инфраструктура как сервис (I**nfrastructure- **as a -S**ervice, **IaaS)**. В традиционной модели хостинга провайдер предоставляет в распоряжение клиента *выделенную* инфраструктуру для того, чтобы клиент выполнял на ней свои приложения. Модель IaaS тоже обеспечивает клиента необходимой инфраструктурой для выполнения приложений, но эта инфраструктура не является выделенной. Она динамически изменяется в зависимости от требований клиента и оплачивается по схеме «оплата за использование». С точки зрения провайдера такая модель может быть реализована на инфраструктуре, способ-

ной гибко реагировать на пики и спады требований каждого клиента. Клиент платит за количество потребленных процессорных циклов, оперативной и дисковой памяти, объем сетевого трафика, но не заботится о физической природе ресурсов — делении диска на разделы, способе увеличения объема памяти, резервировании ресурсов, резервном копировании данных. Провайдер имеет полный контроль над ресурсами инфраструктуры. Клиент, в свою очередь, имеет контроль над тем, какие операционные системы работают на виртуальных машинах инфраструктуры и какие приложения работают под управлением этих ОС. Примером этого типа облачных сервисов является Amazon Web Services. В зависимости от выбранной модели облачных сервисов провайдер и клиент имеют различные зоны контроля над компонентами облачных вычислений (рис. 19.10). Для каждой модели показано распределение зон управления и ответственности между провайдером и клиентом для пяти основных компонентов программно-аппаратного комплекса ИС:

- сетевой инфраструктуры (маршрутизаторы, коммутаторы, линии связи);
- хранилища данных;
- серверов;
- виртуальной машины с операционной системой;
- приложений.

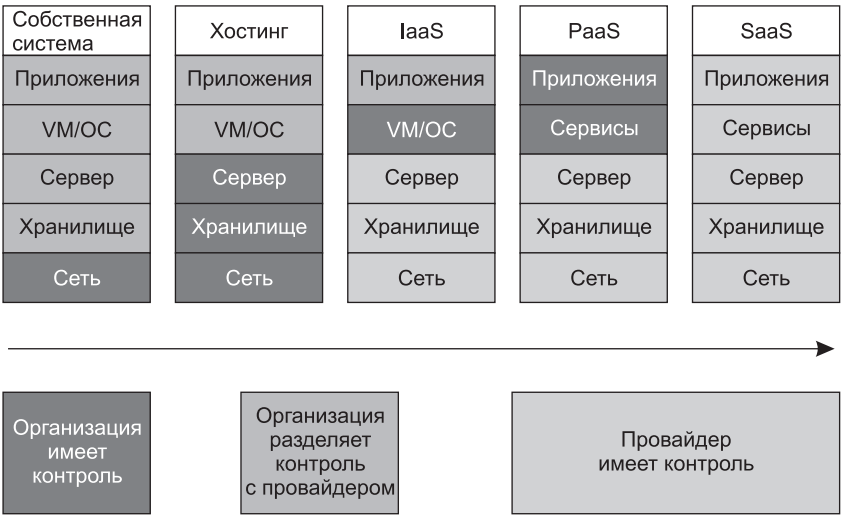


Рис. 19.10. Распределение контроля и ответственности между провайдером и клиентом в разных моделях облачных сервисов

Как видно из рисунка, модель SaaS является крайним случаем, когда провайдер имеет полный контроль над всеми компонентами модели. При использовании услуг модели IaaS клиент имеет контроль над приложениями, работающими в облаке, а провайдер — над тремя нижними слоями модели. Клиент и провайдер разделяют контроль над виртуальной машиной и операционной системой, работающей в среде этой виртуальной машины. Модель PaaS занимает промежуточное положение: здесь провайдер и клиент разделяют

контроль как над средствами виртуализации программного обеспечения (виртуальной машиной, ОС), так и над самими прикладными программами, поскольку хотя они и разрабатываются специалистами предприятия-клиента, в них работают программные модули провайдера, собранные в единую программу по методике провайдера.

ПРИМЕЧАНИЕ

Все три рассмотренные модели облачного сервиса являются развитием популярных моделей хостинга: хостинг аппаратных серверов, хостинг программных веб-серверов и хостинг приложений.

Служба управления сетью

Функции систем управления сетью

Как и любой сложный технический объект, компьютерная сеть требует выполнения различных действий для поддержания ее в рабочем состоянии, анализа и оптимизации ее производительности, защиты от внутренних и внешних угроз. Среди многообразия средств, привлекаемых для достижения этих целей, важное место занимают службы (системы) управления сетью.

Система управления сетью (Network Management System, **NMS**) — это сложный программно-аппаратный комплекс, который контролирует сетевой трафик и управляет коммуникационным оборудованием крупной компьютерной сети.

Системы управления сетью работают, как правило, в *автоматизированном* режиме, выполняя наиболее простые действия автоматически и оставляя человеку принятие сложных решений на основе подготовленной системой информации.

Система управления сетью предназначена для решения следующих групп задач:

- *Мониторинг сети* состоит в постоянном слежении за производительностью, доступностью и безопасностью сети. Для этого система управления сетью получает данные о состоянии и проходящем трафике от различных устройств сети — маршрутизаторов, коммутаторов, серверов и др.
- *Управление конфигурацией сети и именованием* заключается в конфигурировании параметров как отдельных элементов сети, так и сети в целом. Для элементов сети (маршрутизаторы, мультиплексоры и т. п.) конфигурирование состоит в назначении сетевых адресов, идентификаторов (имен), географического положения и пр. Для сети в целом управление конфигурацией обычно начинается с построения карты сети, то есть с отображения реальных связей между элементами сети и связей между ними.
- *Обработка ошибок* включает в себя выявление, определение и устранение последствий сбоев и отказов.
- *Анализ производительности и надежности* связан с оценкой на основе накопленной статистической информации таких параметров, как время реакции системы, пропускная способность реального или виртуального канала связи между двумя конечными абонентами сети, интенсивность трафика в отдельных сегментах и каналах сети, а также вероятность искажения данных при их передаче через сеть. Результаты анализа производительности и надежности позволяют контролировать *соглашение об уровне*

обслуживания (SLA), заключаемое между пользователем сети и ее администраторами (или компанией, продающей услуги). Без средств анализа производительности и надежности поставщик услуг публичной сети или отдел информационных технологий предприятия не сможет ни проконтролировать, ни, тем более, обеспечить нужный уровень обслуживания для конечных пользователей сети.

- *Управление безопасностью* подразумевает контроль доступа к ресурсам сети (данным и оборудованию) и сохранение целостности данных при их хранении и передаче через сеть. Базовыми элементами управления безопасностью являются процедуры аутентификации пользователей, назначение и проверка прав доступа к ресурсам сети, распределение и поддержка ключей шифрования, управление полномочиями и т. п. Часто функции этой группы не включаются в системы управления сетями, а либо реализуются в виде специальных продуктов обеспечения безопасности, например, сетевых экранов или централизованных систем авторизации[1], либо входят в состав операционных систем и системных приложений.
- *Учет работы сети* включает регистрацию времени использования различных ресурсов сети (устройств, каналов и транспортных служб) и ведение биллинговых операций (плата за ресурсы).

В стандартах о системах управления не делается различий между управляемыми объектами, представляющими коммуникационное оборудование (каналы, сегменты локальных сетей, коммутаторы и маршрутизаторы, модемы и мультиплексоры), и объектами, представляющими аппаратное и программное обеспечение компьютеров. Однако на практике деление систем управления по типам управляемых объектов широко распространено.

> В тех случаях, когда управляемыми объектами являются компьютеры, а также их системное и прикладное программное обеспечение, для системы управления часто используют особое название — **система управления системой** (System Management System, **SMS**).

SMS обычно автоматически собирает информацию об установленных в сети компьютерах и создает записи в специальной БД об аппаратных и программных ресурсах. SMS может централизованно устанавливать и администрировать приложения, которые запускаются с серверов, а также удаленно измерять наиболее важные параметры компьютера, ОС, СУБД (например, коэффициент использования процессора или физической памяти, интенсивность страничных прерываний и др.). SMS позволяет администратору брать на себя удаленное управление компьютером в режиме эмуляции графического интерфейса популярных операционных систем.

Архитектура систем управления сетью

Агент управляемого объекта

Для решения перечисленных задач необходимо иметь возможность управления отдельным устройством (объектом). Обычно каждое устройство, требующее достаточно сложного конфигурирования, производитель сопровождает автономной программой конфигурирования и управления, работающей в среде специализированной ОС, установленной

[1] О средствах обеспечения сетевой безопасности читайте в части VII книги.

на этом устройстве. Мы будем называть такой программный компонент **агентом**. Агент взаимодействует с системой управления, передавая ей данные о конфигурации и состоянии устройства и получая от нее команды по управлению этим устройством. Один агент в общем случае может управлять несколькими однотипными устройствами, поддерживая интерфейс с оператором/администратором, который посылает ему запросы и команды на выполнение определенных операций. Агент может выполнять следующие функции:

- хранить, извлекать и передавать информацию о технических и конфигурационных параметрах устройства, включая модель устройства, число портов, тип портов, тип ОС, связи с другими устройствами и др.;
- выполнять, хранить и передавать измерения (подсчеты) характеристик функционирования устройства: число принятых пакетов, число отброшенных пакетов, степень заполнения буфера, состояние порта (рабочее или нерабочее);
- изменять по командам, полученным извне, конфигурационные параметры устройства.

Агент может взаимодействовать с системой управления в соответствии с двумя сценариями:

- в режиме опроса (polling), когда система управления периодически запрашивает от агента данные о состоянии управляемого устройства;
- в режиме потоковой передачи (streaming), когда агент по своей инициативе периодически передает системе управления данные о состоянии управляемого устройства.

Для получения требуемых данных об объекте, выдачи на него управляющих воздействий агент должен иметь возможность взаимодействовать с ним. Многообразие типов управляемых объектов не позволяет стандартизовать способ взаимодействия агента с объектом. Эта задача решается разработчиками при встраивании агентов в коммуникационное оборудование или в ОС. Агент может снабжаться специальными датчиками для получения информации, например датчиками температуры. Агенты могут отличаться разным уровнем интеллекта: от минимального, достаточного лишь для подсчета проходящих через оборудование кадров и пакетов, до весьма высокого, позволяющего выполнять последовательности управляющих команд в аварийных ситуациях, строить временны́е зависимости, фильтровать аварийные сообщения и т. п.

Двухзвенная и трехзвенная схемы управления

Среди задач, определенных для систем управления сетью, есть сравнительно редко выполняемые операции (например, конфигурирование того или иного устройства), а есть и такие, которые требуют частого вмешательства системы (анализ производительности каждого из устройств сети, сбор статистики по загрузке устройств). В первом случае используется ручное управление — администратор со своей консоли передает команды агенту. Понятно, что такой вариант совсем не подходит для глобального мониторинга всех устройств сети. Рассмотрим вначале вариант ручного *двухзвенного* управления (рис. 19.11).

В качестве протокола взаимодействия клиента и сервера может применяться, например, протокол удаленного управления telnet, клиентская часть которого должна быть установлена на компьютере администратора, а серверная — на устройстве. Серверная часть telnet должна также поддерживать интерфейс с агентом, от которого будет поступать информация о состоянии управляемого объекта и значении его характеристик. На клиентской

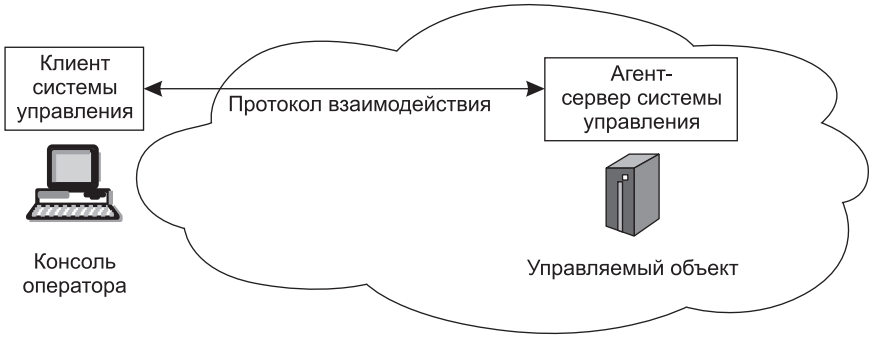


Рис. 19.11. Двухзвенная схема управления устройством

стороне протокол telnet может быть связан с программой поддержки графического пользовательского интерфейса, которая, например, выводит администратору в графическом виде запрашиваемую характеристику. В общем случае администратор может работать с несколькими агентами. В качестве протокола взаимодействия клиентской и серверной частей нередко наряду с telnet используется протокол SSH или протокол веб-службы HTTP.

Для задач, требующих частого выполнения операций управления отдельными устройствами, а также при росте количества управляемых устройств рассмотренная схема уже не может решить поставленную задачу. В схему вводится новое промежуточное звено — менеджер, призванный автоматизировать взаимодействие оператора с множеством агентов. Показанная на рис. 19.12 схема службы управления сетью реализуется в виде *трехзвенного* распределенного приложения, в котором функции между звеньями распределены следующим образом:

- Первое звено — *клиент системы управления*, устанавливается на компьютере оператора, поддерживает пользовательский интерфейс с промежуточным сервером. Клиент системы управления обычно имеет развитый графический интерфейс пользователя (GUI), отображающий топологическую карту сети со всеми ее устройствами и связями между ними, дающую оператору сети наглядную картину состояния сети. При необходимости оператор сети может выделить на карте конкретный узел сети или связь и получить от менеджера подробную информацию о конфигурации узла, его состоянии и статистику проходящего через него трафика.
- Второе звено — промежуточный сервер, который выполняет функции *менеджера* и устанавливается либо на компьютере оператора, либо на специально выделенном компьютере. Менеджер взаимодействует обычно с несколькими клиентами и агентами, обеспечивая диспетчеризацию запросов клиентов к серверам и обрабатывая полученные от агентов данные в соответствии с поставленными перед системой управления задачами. Для повышения надежности и производительности в системе управления может быть предусмотрено несколько менеджеров. В том случае, когда агент работает в режиме потоковой передачи, менеджер часто имеет отдельный компонент, называемый коллектором, который принимает потоки данных от нескольких агентов.
- Третье звено, *агент*, устанавливается на управляемом объекте или связанном с ним компьютере.



Рис. 19.12. Трехзвенная схема управления сетью

Взаимодействие менеджера, агента и управляемого объекта

Остановимся подробнее на той части системы управления, которая относится к взаимодействию менеджера, агента и управляемого объекта (рис. 19.13).

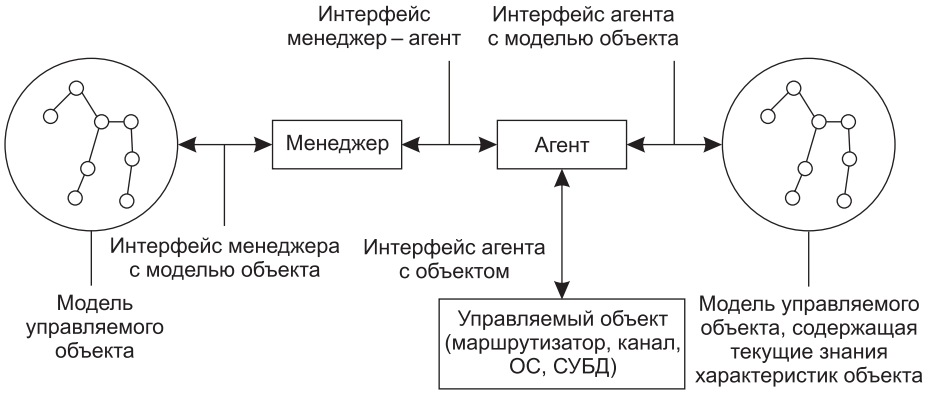


Рис. 19.12. Взаимодействие агента, менеджера и управляемого объекта

Для каждого управляемого объекта в сети создается некоторая *модель объекта*, представляющая все характеристики объекта, необходимые для его контроля. Например, модель маршрутизатора обычно включает такие характеристики, как количество портов, их тип, таблицу маршрутизации, количество кадров и пакетов протоколов канального, сетевого и транспортного уровней, прошедших через эти порты. Модели объектов сети используются менеджером как источник знаний о том, какой набор характеристик имеет тот или иной объект. Модель объекта совпадает с логической схемой базы данных (БД) объекта, хранящей значения его характеристик. Эта БД хранится на устройстве и постоянно пополняется результатами измерений характеристик, которые проводит агент.

Менеджер не имеет непосредственного доступа к базе данных управляемого устройства — для получения конкретных значений характеристик объекта он должен по сети обратиться к его

агенту. Таким образом, агент является посредником между управляемым объектом и менеджером. *Менеджер и агент взаимодействуют по стандартному протоколу*. Этот протокол позволяет менеджеру запрашивать значения параметров, хранящихся в базе данных объекта, а агенту — передавать информацию, на основе которой менеджер должен управлять объектом.

Различают *внутриполосное управление* (in-band), когда команды управления идут по тому же каналу, по которому передаются пользовательские данные, и *внеполосное управление* (out-band), осуществляемое вне канала передачи пользовательских данных. Внутриполосное управление более экономично, так как не требует создания отдельной инфраструктуры передачи управляющих данных. Однако внеполосное управление надежнее, так как соответствующее оборудование может выполнять свои функции даже тогда, когда те или иные сетевые элементы выходят из строя и основные каналы передачи данных оказываются недоступными. Схема «менеджер — агент — управляемый объект» позволяет строить достаточно сложные в структурном отношении системы управления. Наличие нескольких менеджеров позволяет распределить между ними нагрузку по обработке управляющих данных, обеспечивая масштабируемость системы.

Как правило, используются два типа связей между менеджерами: одноранговая (рис. 19.14) и иерархическая (рис. 19.15). Каждый агент, показанный на рисунках, управляет одним или несколькими элементами сети (Network Element, NE), параметры которых он помещает в свою базу данных. Менеджеры извлекают эти параметры из баз данных своих агентов, обрабатывают их и хранят в собственных базах данных. Операторы, работающие за рабочими станциями, могут соединиться с любым из менеджеров и с помощью графического интерфейса просмотреть данные об управляемой сети, а также выдать менеджеру некоторые директивы по управлению сетью или ее элементами.

В случае *одноранговых* связей каждый менеджер управляет своей частью сети на основе информации, получаемой от нижележащих агентов. Центральный менеджер отсутствует, а координация работы менеджеров достигается за счет обмена информацией между базами данных менеджеров. Но одноранговое построение системы управления сегодня считается неэффективным и устаревшим.

Значительно более гибким является *иерархическое* построение связей между менеджерами. Каждый менеджер нижнего уровня выполняет также функции агента для менеджера верхнего уровня. Подобный агент работает уже с укрупненной моделью своей части сети. В базе данных этой модели собирается именно та информация, которая нужна менеджеру верхнего уровня для управления сетью в целом.

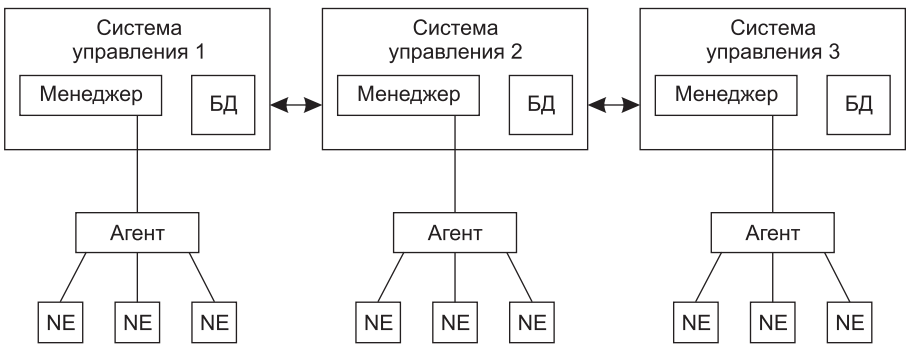


Рис. 19.14. Одноранговые связи между менеджерами

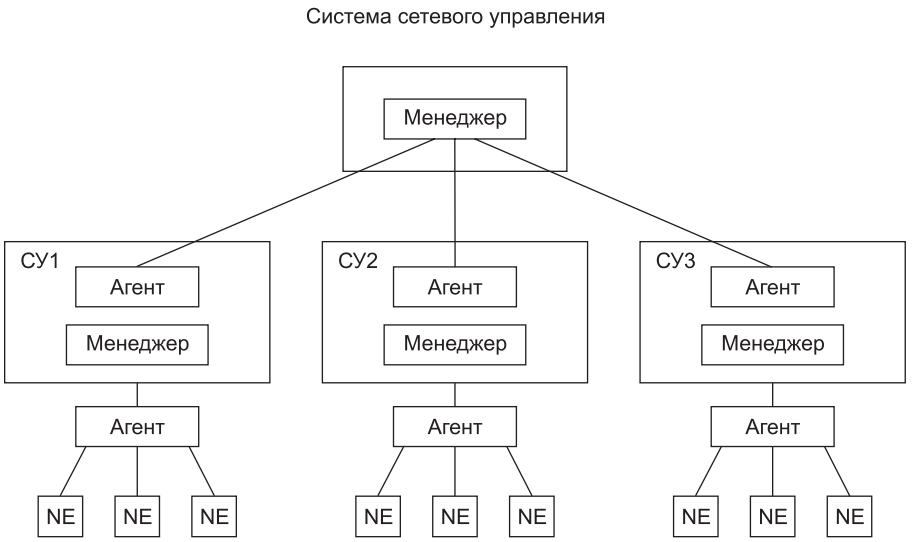


Рис. 19.15. Иерархические связи между менеджерами

Системы управления сетью на основе протокола SNMP

Протокол SNMP

Протокол SNMP (Simple Management Network Protocol — простой протокол сетевого администрирования) используется в качестве стандартного протокола взаимодействия менеджера и агента уже на протяжении более 35 лет. Первая версия этого протокола (RFC 1067) была разработана в 1988 году, а последняя, третья, версия (RFC 3410) — в 2002 году.

Протокол SNMP относится к прикладному уровню стека TCP/IP. Для транспортировки своих сообщений он использует дейтаграммный транспортный протокол UDP, который, как известно, не обеспечивает надежную доставку. Протокол TCP, организующий надежную передачу сообщений на основе соединений, весьма загружает управляемые устройства, которые на момент разработки протокола SNMP были не очень мощные, поэтому от услуг протокола TCP было решено отказаться. SNMP — это протокол типа «запрос-ответ», то есть на каждый запрос, поступивший от менеджера, агент должен передать ответ, иначе говоря агент работает с менеджером в режиме опроса. Особенностью протокола является его чрезвычайная простота — он включает в себя всего несколько команд:

- Команда `GetRequest` используется менеджером для запроса агента о значении какой-либо переменной по ее стандартному имени.
- Команда `GetNextRequest` применяется менеджером для извлечения значения следующего объекта (без указания его имени) при последовательном просмотре таблицы объектов.
- С помощью команды `Response` SNMP-агент передает менеджеру ответ на команду `GetRequest` или `GetNextRequest`.

- Команда `SetRequest` позволяет менеджеру изменять значения какой-либо переменной или списка переменных. С помощью команды `SetRequest` и происходит собственно управление устройством. Агент должен «понимать» смысл значений переменной, которая используется для управления устройством, и на основании этих значений выполнять реальное управляющее воздействие — отключить порт, приписать порт определенной линии VLAN и т. п. Команда `SetRequest` пригодна также для задания условия, при выполнении которого SNMP-агент должен послать менеджеру соответствующее сообщение. Таким образом, может быть определена реакция на такие события, как инициализация агента, рестарт агента, обрыв связи, восстановление связи, неверная аутентификация и потеря ближайшего маршрутизатора. Если происходит любое из этих событий, то агент посылает менеджеру сообщение *по своей инициативе*, используя команду `Trap`. В этом случае агент уже не работает в режиме опроса, но он используется только для очень ограниченного набора аварийных ситуаций и не является основным.
- Команда `GetBulk` позволяет менеджеру получить несколько переменных за один запрос.

SNMP-сообщения, в отличие от сообщений многих других коммуникационных протоколов, не имеют заголовков с фиксированными полями. Любое SNMP-сообщение состоит из трех основных частей: *версии протокола*, *общей строки* и *области данных*.

Общая строка (community string) используется для группирования устройств, управляемых определенным менеджером. Она является своего рода паролем, так как для того, чтобы устройства могли взаимодействовать по протоколу SNMP, они должны иметь одно и то же значение этого идентификатора (по умолчанию часто употребляется строка «public»). Однако этот механизм служит скорее для распознавания партнеров, нежели для безопасности.

В области данных содержатся описанные команды протокола, а также имена объектов и их значения. Область данных состоит из одного или более блоков, каждый из которых может относиться к одному из перечисленных типов команд протокола SNMP. Для каждого типа команды определен свой формат. Например, формат блока, относящегося к команде `GetRequest`, включает следующие поля:

- идентификатор запроса;
- статус ошибки (есть или нет);
- индекс ошибки (тип ошибки, если она есть);
- список имен объектов SNMP MIB, включенных в запрос.

База данных MIB

База данных MIB (Management Information Base) содержит значения множества различных типов переменных, характеризующих конкретный управляемый объект. В самой первой версии стандарта (MIB v1) для характеристики устройства предлагалось использовать 114 типов переменных. Эти переменные организованы в виде дерева. Из корня выходит восемь ветвей, соответствующих следующим восьми группам переменных:

- *System* — общие данные об устройстве (например, идентификатор поставщика, время последней инициализации системы);
- *Interfaces* — параметры сетевых интерфейсов устройства (например, их количество, типы, скорости обмена, максимальный размер пакета);

- *Address Translation Table* — описание соответствия между сетевыми и физическими адресами (например, по протоколу ARP);
- *Internet Protocol* — данные, относящиеся к протоколу IP (адреса IP-шлюзов, хостов, статистика об IP-пакетах);
- *ICMP* — данные, относящиеся к протоколу ICMP;
- *TCP* — данные, относящиеся к протоколу TCP (число переданных, принятых и ошибочных TCP-сообщений);
- *UDP* — данные, относящиеся к протоколу UDP (число переданных, принятых и ошибочных UPD-дейтаграмм);
- *EGP* — данные, относящиеся к протоколу EGP (число принятых с ошибками и без ошибок сообщений).

Каждая группа характеристик образует отдельное поддерево.

Версия SNMP MIB 2 позволяет строить более гибкие модели управляемых устройств за счет новых типов данных (например, 64-битных указателей), поддерживает режим групповой передачи данных (нужный для передачи табличных данных), а также улучшает безопасность систем управления за счет поддержки нескольких механизмов аутентификации вместо примитивного механизма аутентификации общей строки в SNMP MIB v1.

Модели SNMP MIB описываются и используются менеджерами и управляемыми устройствами в виде текстовых файлов, эти файлы можно читать любым текстовым редактором. Для понимания функциональности некоторой MIB нужно иметь в виду, что ее текст следует синтаксису документа «Структура управляющей информации» (Structure of Management Information Version 2, SMIv2, RFC 2578).

Для учета специфики конкретного сетевого устройства — маршрутизатора, коммутатора, модема и т. д. — производители часто выпускают свои собственные, фирменные MIB, которые расширяют возможности сбора данных о состоянии этого устройства.

Режим удаленного управления и протокол telnet

Режим удаленного управления, называемый также режимом **терминального доступа**, предполагает, что пользователь превращает свой компьютер в виртуальный терминал другого компьютера, к которому он получает удаленный доступ.

В период становления компьютерных сетей, то есть в 70-е годы, поддержка такого режима была одой из главных функций сети. Устройства PAD сетей X.25 существовали именно для того, чтобы обеспечить удаленный доступ к мейнфреймам для пользователей, находившихся в других городах и работавших за простыми алфавитно-цифровыми терминалами.

Режим удаленного управления реализуется специальным протоколом прикладного уровня, работающим поверх транспортных протоколов, которые связывают удаленный узел с компьютерной сетью. Существует большое количество протоколов удаленного управления — как стандартных, так и фирменных. Для IP-сетей наиболее старым протоколом этого типа является telnet (RFC 854).

Протокол **telnet** работает в архитектуре клиент-сервер, обеспечивая эмуляцию алфавитно-цифрового терминала и ограничивая пользователя режимом командной строки.

При нажатии клавиши соответствующий код перехватывается клиентом telnet, помещается в TCP-сообщение и отправляется через сеть узлу, которым пользователь хочет управлять. При поступлении на узел назначения код нажатой клавиши извлекается из TCP-сообщения сервером telnet и передается ОС узла, которая рассматривает сеанс telnet как один из сеансов локального пользователя. Если ОС реагирует на нажатие клавиши выводом очередного символа на экран, то для сеанса удаленного пользователя этот символ также упаковывается в TCP-сообщение и по сети отправляется удаленному узлу. Клиент telnet извлекает символ и отображает его в окне своего терминала, эмулируя терминал удаленного узла.

Протокол telnet был реализован в среде Unix и, наряду с электронной почтой и FTP-доступом к архивам файлов, был популярным сервисом интернета. Однако, поскольку для аутентификации пользователей в технологии telnet применяются пароли, передаваемые через сеть в виде обычного текста (что означает, что пароли могут быть легко перехвачены и использованы), telnet сейчас работает преимущественно в пределах одной локальной сети, где возможностей для перехвата пароля гораздо меньше. Для удаленного управления узлами через интернет вместо telnet обычно применяется протокол SSH (Secure SHell), который, как и telnet, был первоначально разработан для ОС Unix[1]. SSH, как и telnet, передает набираемые на терминале пользователя символы на удаленный узел без интерпретации их содержания. Однако в SSH предусмотрены меры по защите передаваемых аутентификационных и пользовательских данных.

Сегодня основной областью применения telnet является управление не компьютерами, а коммуникационными устройствами: маршрутизаторами, коммутаторами и хабами. Таким образом, он уже скорее представляет собой не пользовательский протокол, а протокол администрирования, то есть альтернативу SNMP.

Тем не менее отличие между протоколами telnet и SNMP принципиально. Telnet предусматривает обязательное участие человека в процессе администрирования, так как, по сути, лишь транслирует команды, которые вводит администратор при конфигурировании или мониторинге маршрутизатора (иного коммуникационного устройства). Протокол SNMP, наоборот, рассчитан на автоматические процедуры мониторинга и управления, хотя и не исключает возможности участия администратора в этом процессе. Для устранения опасности, порождаемой передачей паролей в открытом виде через сеть, коммуникационные устройства усиливают степень своей защиты. Обычно применяется многоуровневая схема доступа, когда открытый пароль дает возможность только чтения базовых характеристик конфигурации устройства, а доступ к средствам изменения конфигурации требует другого пароля, который уже не передается в открытом виде.

Потоковая телеметрия

Потоковая телеметрия (streaming telemetry) стала в последнее время все чаще применяться в системах управления сетями вместо протокола SNMP и баз SNMP MIB.

Системы управления сетью, работающие на основе протокола SNMP, обладают рядом недостатков, что неудивительно, принимая во внимание тот факт, что этот протокол был

[1] См. материал «Аутентификация и авторизация в операционных системах» на сайте книги.

разработан в расчете на маршрутизаторы 80-х годов с очень ограниченными по мощности (по сегодняшним меркам) процессорами.

Основными недостатками таких систем являются следующие:

- Метод опроса *нерационален в отношении использования процессорной мощности агента и менеджера*, так как он не допускает передачу данных о состоянии управляемого устройства только при изменениях состояния, а это приводит к большому количеству неинформативных передач, несущих повторяющиеся данные. Метод опроса плохо масштабируется — нагрузка на менеджер растет пропорционально количеству опрашиваемых устройств.
- *Ненадежный транспорт UDP*. При возникновении аварийной ситуации сообщение Trap может быть утеряно, что крайне нежелательно.
- Недостаточное разрешение данных о состоянии устройства по времени. Для уменьшения нагрузки на процессор устройства эти данные обновляются в MIB с большим периодом, обычно 5 минут, так что менеджер может пропустить важные кратковременные тренды.
- Недостаточно гибкая модель управляемого устройства. MIB имеет простую структуру в виде дерева и не позволяет отражать сложные зависимости между параметрами устройства.

Потоковая телеметрия преодолевает эти недостатки за счет:

- Отказа от передачи данных по запросу менеджера. Вместо этого управляемое устройство генерирует поток данных о своем состоянии и направляет его **коллектору** — специальной системе, которая собирает данные от управляемых устройств и помещает их в базу данных, доступную менеджеру. Передача инициативы устройству позволяет более рационально использовать мощность его процессора, что уменьшает период генерации данных до секундного, а иногда и субсекундного диапазона. Кроме того, возможна генерация данных при соблюдении некоторого условия — например, данные о передаваемом интерфейсом трафике могут начать передаваться только после превышения порога, а эта возможность резко уменьшает объем передаваемых малоинформативных данных.
- Использования гибких и расширяемых моделей на основе языка YANG (Yet Another Next Generation). YANG является стандартом IETF (RFC 7950).
- Обмена данными между устройством и коллектором с помощью протокола gRPC (Google Remote Procedure Call). Этот протокол изначально был разработан компанией Google для своих внутренних нужд, но быстро стал популярен в сообществе программистов из-за его высокой эффективности при передаче потоковых данных в реальном времени. Протокол gRPC работает на прикладном уровне модели OSI поверх протоколов HTTP/TCP, а значит, обеспечивает не только быструю, но и надежную передачу данных. Сегодня протокол gRPC является протоколом с открытым кодом и поддерживается организацией Cloud Native Computing Foundation (CNCF).

Идея потоковой телеметрии обсуждалась с самого начала работ по стандартизации систем управления сетями в 1980-е годы, но не нашла поддержки из-за сложности ее реализации сетевыми устройствами того времени. В результате маршрутизаторы, коммутаторы и другие сетевые устройства стали поддерживать потоковую телеметрию только в конце 2000-х — начале 2010-х годов.

ГЛАВА 20 MPLS — транспортная технология глобальных сетей

Принципы работы виртуального канала

В этом разделе мы дополним[1] ваши знания о принципах работы виртуального канала. В сети с виртуальными каналами два узла могут начать обмен данными только после того, как между ними будет установлено логическое соединение — **виртуальный канал**. Виртуальный канал лучше защищает пользователей от внешних атак, поскольку у злоумышленника нет возможности посылать пакеты данных от одного произвольного узла к другому, что вполне можно сделать в сети, построенной на транспортной технологии дейтаграммного типа: IP или Ethernet. Продвижение кадров вдоль виртуального канала происходит не на основе адресов конечных узлов, а на основе *метки*, позволяющей коммутаторам сети определять принадлежность кадров тому или иному виртуальному каналу и продвигать их соответственно. Значение метки потока изменяется в каждом коммутаторе при передаче кадра с входного интерфейса на выходной — в этом случае говорят, что происходит коммутация по меткам. Коммутация по меткам позволяет избавиться от требования уникальности их значений в пределах сети, которую обеспечить сложно; чтобы кадры различных виртуальных каналов не смешивались, достаточно обеспечить уникальность значений меток только в пределах отдельного интерфейса. Из-за того что коммутация по меткам является неотъемлемым атрибутом технологий виртуальных каналов, у них есть и второе название — **технологии коммутации по меткам**.

Технически установление виртуального канала означает формирование записей в *таблицах продвижения кадров* на каждом коммутаторе вдоль виртуального канала. Такая таблица включает информацию о продвижении — на какой выходной порт нужно передать кадр с данной меткой, какое новое значение нужно присвоить метке после передачи кадра на выходной интерфейс.

На рис. 20.1 показан фрагмент сети, состоящей из двух коммутаторов, S1 и S2, и четырех конечных узлов, C1–C4. Через эти коммутаторы проложено три виртуальных канала: C1–C2, C1–C4 и C3–C4.

Эти каналы являются *двунаправленными* — кадры могут передаваться по ним в любом из двух направлений. Для каждого виртуального канала в таблице продвижения имеется две записи — по одной для каждого направления. Например, первая запись в таблице коммутации коммутатора *S*1 (запись 1-101-2-103) определяет работу коммутатора по продвижению кадров виртуального канала *C*1–*C*2 в направлении от *C*1 к *C*2; она предписывает комму-

[1] См. главу 3.

татору *S1* передать кадр, принятый на порт 1 со значением метки 101, на порт 2, поменяв значение метки (скоммутировать метку) на 103. Третья запись (2-103-1-101) означает, что все пакеты, которые поступят на порт 2 со значением метки 102, будут продвигаться на порт 3, а ее значение поменяется на 101.

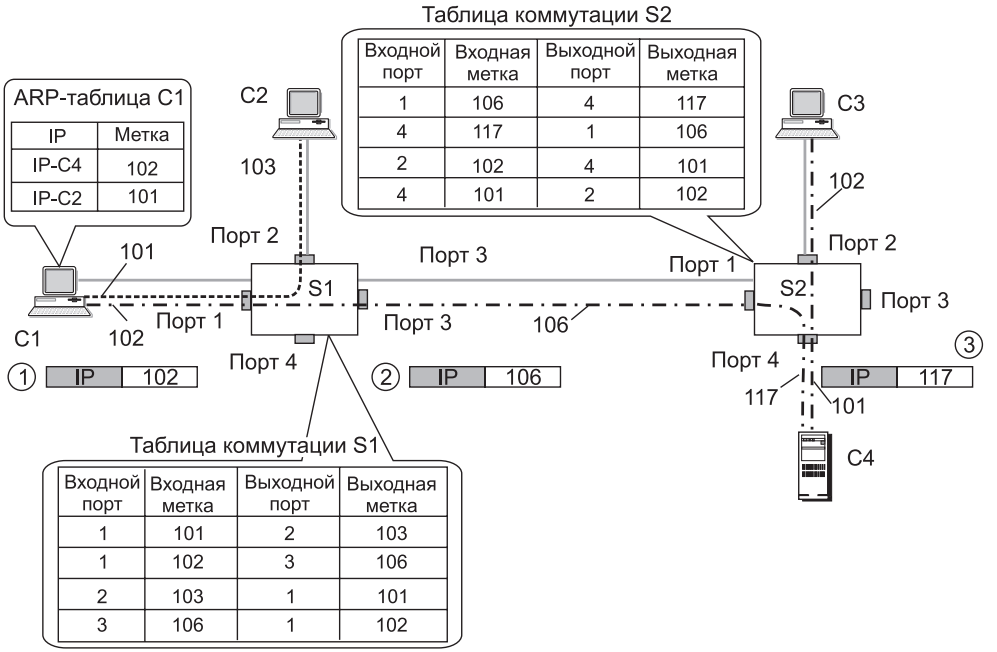


Рис. 20.1. Продвижение кадров вдоль виртуальных каналов

Существуют также *однонаправленные* виртуальные каналы. В случае их использования для дуплексного обмена информацией нужно установить два независимых виртуальных канала между конечными узлами, по одному для каждого направления. Виртуальные каналы делятся на два класса:

- **коммутируемые виртуальные каналы** (Switched Virtual Circuit, **SVC**);
- **постоянные виртуальные каналы** (Permanent Virtual Circuit, **PVC**).

Создание коммутируемого виртуального канала происходит по инициативе конечного узла сети с помощью специального протокола, посылающего пакет с запросом на установление соединения в направлении к узлу назначения виртуального канала. Название «коммутируемый» отражает тот факт, что канал создается динамически по требованию узла-отправителя аналогично установлению коммутируемого соединения в телефонной сети. Для поддержания режима SVC в сети должны существовать таблицы маршрутизации, в соответствии с которыми продвигается пакет с запросом соединения. По отношению к *пакету с запросом соединения* сеть работает в дейтаграммном режиме, и такой пакет должен содержать *адрес назначения конечного узла, а не метку*.

Постоянный виртуальный канал устанавливается вручную. Администратор создает его на достаточно длительное время (отсюда название канала), возможно, с привлечением

централизованной системы управления сетью. Пограничный коммутатор сети принимает пакеты от внешней сети, которая может и не поддерживать технику виртуальных каналов. Пограничный коммутатор должен каким-то образом отображать приходящие извне пакеты на один из виртуальных каналов сети. В простейшем случае такое отображение (mapping) выполняется на основе входного физического интерфейса, то есть все кадры, приходящие на некоторый входной интерфейс, отображаются на один и тот же виртуальный канал. В более сложных случаях необходимо различать несколько потоков, приходящих на входной интерфейс, и отображать их на разные виртуальные каналы. В таком случае в пограничном коммутаторе наряду с таблицей продвижения должна существовать таблица отображения потоков. В примере рис. 20.1 такая таблица имеется у конечного узла *C*1. В ней в качестве признака потока используются IP-адреса назначения, поэтому таблица отображения представляет собой ARP-таблицу.

Виртуальные каналы обычно имеют *двухточечную топологию*. Но существуют каналы и с другим типом топологии — *звезда* (рис. 20.2). В таком канале один и тот же кадр передается от источника — центра звезды, называемого также концентратором, — вдоль лучей

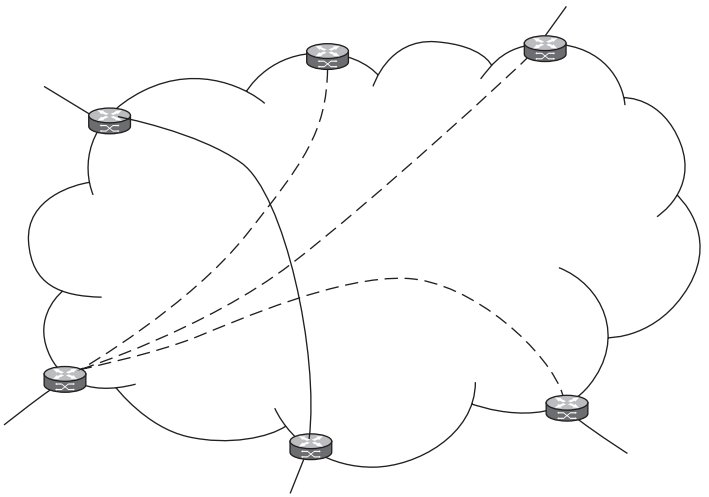


Рис. 20.2. Топологии виртуальных каналов

всем конечным узлам. Конечные узлы не могут использовать виртуальный канал звездообразной топологии для обмена кадрами между собой, который передает кадры в обратном направлении только от конечного узла к центральному узлу. Виртуальные каналы со звездообразной топологией рассчитаны на эффективную поддержку группового вещания.

Виртуальный канал является удобным инструментом для *инжиниринга трафика*

Это объясняется тем, что он может быть установлен независимо в каждом промежуточном коммутаторе путем соответствующего назначения локальных меток, выполняемого

администратором сети или внешней программной системой. Поток пакетов, который должен быть передан по виртуальному каналу, может быть определен гибко и с любой степенью детализации, в этом определении могут использоваться не только IP-адреса назначения, как это происходит в IP-сетях, но и любые признаки: IP-адреса источника, TCP/UDP-порты назначения, поле DSCP и т. п., что также повышает эффективность инжиниринга трафика.

Сети, работающие на основе техники виртуальных каналов, относятся к типу **сетей, не поддерживающих широковещание с множественным доступом** (Non Broadcast Multiple Access, **NBMA**). Действительно, у такой сети существует произвольное количество конечных узлов, но отсутствует возможность послать кадр сразу всем узлам — ни двухточечный, ни звездообразный виртуальный канал это не позволяет. В сетях NBMA протокол IP не может воспользоваться услугами протокола ARP для автоматического построения ARP-таблицы — эти услуги основаны на широковещательных запросах. В тех случаях, когда входящий поток отображается на виртуальный канал на основе IP-адреса, таблицу отображения, которая является здесь ARP-таблицей, приходится строить вручную или же с помощью некоторого дополнительного протокола, не использующего широковещание.

Эффективность виртуальных каналов

Сравнение эффективности виртуальных каналов мы проведем отдельно для коммутируемых и постоянных виртуальных каналов, сопоставляя первые с дейтаграммными технологиями, а вторые — с выделенными физическими каналами.

Применение *коммутируемых виртуальных каналов* требует предварительного установления соединения, что вносит дополнительную задержку перед передачей данных по сравнению с применением дейтаграммных протоколов. Эта задержка особенно сказывается при передаче небольшого объема данных, когда время установления виртуального канала может быть соизмеримым со временем передачи данных. Кроме того, дейтаграммный метод быстрее адаптируется к изменениям в сети. При отказе коммутатора или линии связи вдоль виртуального канала соединение разрывается, после чего виртуальный канал нужно прокладывать заново, обходя отказавшие участки сети.

Однако сравнивая эти два принципиально различных подхода, следует учесть, что время, затраченное на установление виртуального канала, компенсируется последующей быстрой передачей всего потока пакетов. Маршрутизация пакетов в сети с поддержкой виртуальных каналов ускоряется за счет двух факторов. Первый состоит в том, что решение о продвижении пакета принимается быстрее, так как таблица коммутации, в которой есть информация только об установленных виртуальных каналах, чаще всего существенно меньше таблицы маршрутизации, в которой число записей определяется количеством сетей назначения (размер таблицы маршрутизации магистральных IP-маршрутизаторов провайдеров интернета составлял в 2025 году около 1 000 000 записей). Вторым фактором является уменьшение доли служебной информации в пакетах. Адреса конечных узлов в глобальных сетях обычно имеют достаточно большую длину — 4 байта в версии IPv4, 16 байт в версии IPv6, MAC-адрес имеет длину 6 байт. Номер же виртуального канала обычно занимает 10–12 бит, так что накладные расходы на адресную часть существенно сокращаются, а значит, полезная скорость передачи данных возрастает.

Постоянные виртуальные каналы являются гораздо более эффективными в отношении производительности передачи данных, чем коммутируемые. Значительную часть работы по маршрутизации пакетов сети выполняет администратор, вручную прокладывая постоянные виртуальные каналы и оставляя коммутаторам только продвижение пакетов на основе готовых таблиц коммутации портов.

Постоянный виртуальный канал подобен выделенному физическому каналу в том смысле, что для каждой операции обмена данными не требуется заново устанавливать или разрывать соединение. Отличие — в том, что пользователь PVC не имеет гарантий относительно действительной пропускной способности канала. Зато применение PVC обычно намного дешевле, чем аренда выделенной линии, так как пользователь делит пропускную способность сети с другими пользователями.

Постоянные виртуальные каналы выгодно использовать для передачи *агрегированных потоков трафика*, состоящих из большого количества индивидуальных потоков абонентов сети. В этом случае виртуальный канал прокладывается не между конечными абонентами, а между участком магистрали сети, на котором данный агрегированный поток существует, например от одного пограничного маршрутизатора сети оператора связи до другого. В силу закона больших чисел агрегированные потоки обладают высокой степенью устойчивости, так что для них нет смысла динамически создавать коммутируемые виртуальные каналы — лучше эффективно использовать постоянные, которые при хорошем планировании (методами инжиниринга трафика) оказываются достаточно загруженными.

Подводя итог, можно сказать, что виртуальные каналы более эффективны при передаче долговременных, чем кратковременных потоков, так как в этом случае снижаются удельные затраты на установление соединений.

Техника виртуальных каналов использовалась во многих технологиях глобальных компьютерных сетей прошлого — X.25, Frame Relay, ATM. Эти технологии сегодня не применяются не потому, что сама идея виртуального канала не оправдала себя, а из-за их недостаточной интеграции с протоколами стека TCP/IP. Описания этих технологий можно найти на сайте книги.

(S) *Технология X.25*

(S) *Технология Frame Relay*

(S) *Технология ATM*

Базовые принципы и механизмы MPLS

Технология **многопротокольной коммутации по меткам** (MultiProtocol Label Switching, **MPLS**) считается многими специалистами одной из самых эффективных транспортных технологий, объединяющих технику виртуальных каналов с функциональностью стека TCP/IP. Главное достоинство MPLS видится сегодня многим специалистам в способности предоставлять разнообразные транспортные услуги в IP-сетях, в первую очередь — услуги виртуальных частных сетей. Эти услуги отличаются разнообразием: они могут предоставляться как на сетевом, так и на канальном уровне. Кроме того, MPLS дополняет дейтаграммные IP-сети таким важным свойством, как передача трафика в соответствии с техникой виртуальных каналов, что позволяет выбирать нужный режим передачи тра-

фика в зависимости от требований услуги. Виртуальные каналы MPLS обеспечивают инжиниринг трафика, так как они поддерживают детерминированные маршруты.

Совмещение коммутации и маршрутизации

Технология MPLS объединяет в одном коммуникационном устройстве коммутацию виртуальных каналов с IP-маршрутизацией.

Дейтаграммное продвижение реализуется протоколом IP — он работает точно так же, как и в традиционном IP-маршрутизаторе, при этом таблица маршрутизации может создаваться как вручную, так и протоколами маршрутизации стека TCP/IP. В то же время в этом коммуникационном устройстве, называемом **маршрутизатором с коммутацией по меткам** (Label Switch Router, **LSR**)[1], имеется второй модуль продвижения, работающий в соответствии с техникой коммутации виртуальных каналов, который здесь называется модулем **коммутации по меткам** (рис. 20.3). Метки в технологии MPLS имеют ***локальное*** значение: как и в других технологиях, использующих технику виртуальных каналов, метки уникальны только в пределах одного интерфейса одного узла сети. Здесь нужно сделать оговорку: существуют особые режимы работы технологии MPLS, например, сегментная маршрутизация, использующая наряду с локальными метками и глобальные метки, но это исключение из общего правила мы будем оговаривать особо.

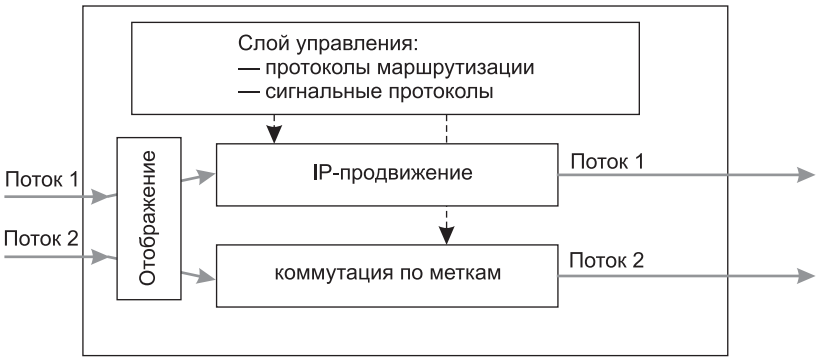


Рис. 20.3. Структура LSR

Оба модуля продвижения управляются одним и тем же слоем управления LSR, куда наряду с традиционными протоколами IP-маршрутизации, такими как RIP, OSPF, IS-IS и BGP, входят и новые протоколы, называемые сигнальными. **Сигнальные протоколы** нужны для автоматического установления в сети виртуального пути, называемого в технологии MPLS **путем коммутации по меткам** (Label Switching Path, **LSP**). Наличие общего слоя управления позволяет LSR гибко использовать два модуля продвижения — одну часть потоков данных он может продвигать, применяя технику IP-продвижения, а другую — с помощью техники коммутации по меткам. Слой управления имеет информацию о топологии сети, необходимую для работы каждого уровня продвижения.

---

[1] Заметим, что на практике такое устройство чаще всего по-прежнему называют IP-маршрутизатором или IP/MPLS-маршрутизатором.

Какие именно потоки нужно продвигать тем или иным способом, решает администратор LSR, который конфигурирует его соответствующим образом. В зависимости от параметров конфигурации, IP/MPLS-маршрутизатор выполняет *отображение* входных потоков пакетов на модуль IP-продвижения или модуль коммутации по меткам. Поток выделяется в соответствии с его признаками, к которым могут относиться IP-адреса, MAC-адреса, порты TCP/UDP и другие поля заголовка пакета и кадра, обычно используемые для классификации потоков данных.

Пути коммутации по меткам прокладываются в сети независимо от того, существует ли поток пакетов в сети в данное время или только является *топологически* возможным. Последнее означает, что в сети имеются некоторые два конечных узла, определяемые IP-адресами, и между ними есть возможность установить путь через промежуточные IP/MPLS-маршрутизаторы. Такое свойство путей коммутации по меткам иногда называют «топологически ведомым» (topology-driven). Здесь же нужно отметить, что при разработке технологии MPLS обсуждалась и другая идея — идея установления путей коммутации по меткам в зависимости от характеристик существующих в сети потоков. Такой способ установления путей коммутации по меткам можно было бы назвать «ведомым данными» (data-driven). Из сравнения свойств дейтаграммных технологий и технологий виртуальных каналов читателям известно, что серьезным недостатком коммутируемых виртуальных путей является их неэффективность при передаче кратковременных потоков из-за внесения задержки при их установлении. Имея в виду этот недостаток, сторонники техники «ведомых данными» путей коммутации по меткам (MPLS) предлагали динамически создавать такие пути только для долговременных потоков (то есть только после того, как некоторый поток покажет свою «долговременность»), а пакеты кратковременных потоков передавать путем стандартного IP-продвижения, эффективного для этого типа потоков. Но такое кардинальное изменение логики работы коммутационных устройств, требующее измерения длительности существования потока, было, в конце концов, отвергнуто и заменено более привычным способом формирования таблиц продвижения заранее (до появления потока) в соответствии с топологией сети.

В заключение обзора идеи совмещения техники дейтаграммной передачи и виртуальных каналов вновь подчеркнем значение *единого слоя управления* в устройстве LSR. Если считать, что на канальном уровне коммутаторы поддерживают технологию MPLS, то чем такая двухслойная структура сети отличается от двухслойной структуры устройства LSR? И почему с помощью двухслойной сети нельзя оказывать те же услуги и с таким же удобством для оператора сети, что и с помощью однослойной внешне, но двухслойной внутренне сети устройств LSR? Основное отличие как раз и заключается в том, что в случае двухслойной сети каждый слой продвижения данных находится под контролем собственного слоя управления, созданного без учета существования друг друга в одной сети, поэтому заставить их работать вместе и скоординированно непросто. Скорее всего, администратор сети сможет использовать факт наличия в сети двух слоев, способных предоставлять транспортные услуги, простым физическим подключением различных клиентов ко входным интерфейсам разных слоев.

Наличие единого слоя управления в устройствах LSR как раз и создает возможность координировать работу двух разных слоев продвижения пакетов из одного центра. Протоколы маршрутизации и сигнализации в этом случае явно учитывают существование двух способов продвижения пакетов, при этом как автоматически, так и с учетом конфигурации, заданной администратором, обслуживают потоки данных пользователей наиболее рациональным способом.

Пути коммутации по меткам

Пути коммутации по меткам в технологии MPLS представляют собой некоторый гибрид коммутируемых и постоянных виртуальных каналов. Их можно отнести к коммутируемым, так как они устанавливаются в сети автоматически с помощью сигнальных протоколов. В то же время они могут считаться постоянными, так как их создание инициируется не динамическим запросом, вызванным необходимостью установить сеанс связи между конечными узлами и передать некоторые данные, а имеющейся топологией сети, мало изменяющейся во времени. Пути коммутации по меткам в технологии MPLS поддерживают инжиниринг трафика за счет соответствующих протоколов маршрутизации и специального сигнального протокола. Рассмотрим работу пути LSP на примере сети, показанной на рис. 20.4. Эта MPLS-сеть взаимодействует с несколькими IP-сетями, возможно, не поддерживающими технологию MPLS.

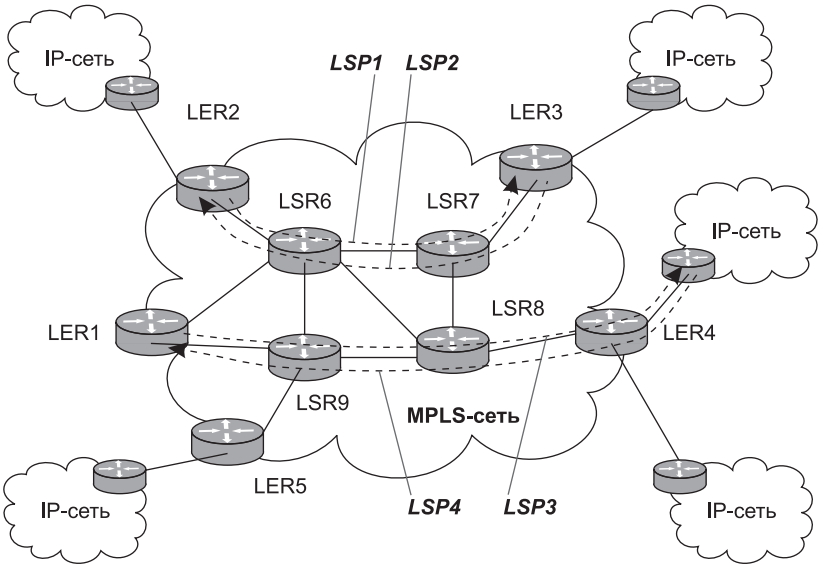


Рис. 20.4. MPLS-сеть

На рисунке мы видим новый тип устройств — это пограничные устройства LSR, которые в технологии MPLS обозначаются как «пограничные коммутирующие по меткам маршрутизаторы» (Label switch Edge Router, **LER**). Устройство LER, являясь функционально более сложным, принимает трафик от других сетей в форме стандартных IP-пакетов, а затем добавляет к каждому пакету метку и направляет вдоль соответствующего пути к выходному устройству LER через несколько промежуточных устройств LSR. При этом пакет продвигается не на основе IP-адреса назначения, а на основе метки. Как и в других технологиях, использующих технику виртуальных каналов, метка имеет локальное значение в пределах каждого устройства LER или LSR, то есть при передаче пакета с входного интерфейса на выходной выполняется смена значения метки.

При принятии решения о выборе следующего хопа блок продвижения по меткам использует *таблицу коммутации*, которая в стандарте MPLS носит название **таблицы продвиже-**

ния LFIB (Label Forwarding Information Base) и похожа на аналогичные таблицы других технологий, основанных на технике виртуальных каналов (табл. 20.1).

Таблица 20.1. Пример таблицы продвижения в технологии MPLS

Входной интерфейс	Метка	Следующий хоп	Действия
S0	245	S1	256
S0	27	S2	45
...	...	...	...

Внимательный читатель заметил, наверное, небольшое отличие данной таблицы от обобщенной таблицы коммутации, представленной на рис. 20.1. Действительно, вместо поля выходного интерфейса здесь поле следующего хопа, а также добавлено новое поле — поле действий. В данном примере это действие — Swap, то есть замена входной метки на выходную, — аналогично действию по умолчанию таблицы обобщенной коммутации. Но в MPLS определены и другие действия над метками, мы увидим это далее.

Рассматриваемые таблицы для каждого устройства LSR формируются *сигнальным протоколом.* В MPLS используется два различных сигнальных протокола: **протокол распределения меток** (Label Distribution Protocol, **LDP**) и модификация уже знакомого нам протокола резервирования ресурсов RSVP. Формируя таблицы продвижения на устройствах LSR, сигнальный протокол прокладывает через сеть виртуальные маршруты, которые в технологии MPLS называют **путями коммутации по меткам** (Label Switching Path, **LSP**).

В том случае, когда метки устанавливаются в таблицах продвижения с помощью протокола LDP, маршруты виртуальных путей LSP совпадают с маршрутами IP-трафика, так как они выбираются обычными протоколами маршрутизации стека TCP/IP. Модификация протокола RSVP, изначально разработанного для резервирования параметров QoS (см. раздел «Работа в недогруженном режиме» главы 5), используется для прокладки путей, выбранных в соответствии с техникой инжиниринга трафика, поэтому эта версия протокола получили название RSVP TE (Traffic Engineering). Метки могут также использоваться в технике маршрутизации от источника, например SR-MPLS. В этом случае они переносятся в заголовке MPLS-кадра. Эта техника поддерживает инжиниринг трафика. Можно также формировать таблицы MPLS-продвижения вручную, создавая там статические записи, подобные статическим записям таблиц маршрутизации.

LSP представляет собой *однонаправленный* виртуальный канал, поэтому для передачи трафика между двумя устройствами LER нужно установить по крайней мере два пути коммутации по меткам — по одному в каждом направлении. На рис. 20.4 показаны две пары путей коммутации по меткам, соединяющие устройства LER2 и LER3, а также LER1 и LER4. LER выполняет такую важную функцию, как направление входного трафика в один из исходящих из LER путей LSP. Для реализации этой функции в MPLS введено понятие **класса эквивалентности продвижения** (Forwarding Equivalence Class, **FEC**).

Класс эквивалентности продвижения — это группа IP-пакетов, имеющих одни и те же требования к условиям транспортировки (транспортному сервису). Все пакеты, принадлежащие к данному классу, продвигаются через MPLS-сеть по одному виртуальному пути LSP.

Входящий пакет относят к тому или иному классу на основании некоторых признаков. Вот несколько примеров классификации:

- *На основании IP-адреса назначения.* Это наиболее близкий к принципам работы IP-сетей подход, который состоит в том, что для каждого префикса сети назначения, имеющегося в таблице LER-маршрутизации, создается отдельный класс FEC. Протокол LDP, который мы далее рассмотрим, полностью автоматизирует процесс создания классов FEC по этому способу.
- *В соответствии с требованиями инжиниринга трафика.* Классы выбираются таким образом, чтобы добиться баланса загрузки каналов сети.
- *В соответствии с требованиями VPN.* Для конкретной виртуальной частной сети клиента создается отдельный класс FEC.
- *По типам приложения.* Например, трафик IP-телефонии (RTP) составляет один класс FEC, а веб-трафик — другой.
- *По интерфейсу, с которого получен пакет.*
- *По MAC-адресу назначения кадра*, если это кадр Ethernet.

Как видно из примеров, при классификации трафика в MPLS могут использоваться признаки, не только взятые из заголовка IP-пакета, но и многие другие, включая информацию канального (MAC-адрес) и физического (интерфейс) уровней.

После принятия решения о принадлежности пакета к определенному классу FEC его нужно связать с существующим путем LSP. Для этой операции LER использует таблицу FTN (FEC To Next hop — отображение класса FEC на следующий хоп). Таблица 20.2 представляет собой пример FTN.

Таблица 20.2. Пример таблицы FTN

Признаки FEC	Метка
123.20.0.0/16; 195.14.0.0/16	106
194.20.0.0/24; eth1	107

На основании таблицы FTN каждому входящему пакету назначается соответствующая метка, после чего этот пакет становится неотличим в домене MPLS от других пакетов того же класса FEC: все они продвигаются по одному и тому же пути внутри домена. У администратора сети имеется возможность формировать таблицы FEC или же корректировать их, если они формируются автоматически.

Сложная настройка и конфигурирование выполняются только в LER, а все промежуточные устройства LSR делают простую работу, продвигая пакет в соответствии с техникой виртуального канала. Выходное устройство LER удаляет метку и передает пакет в следующую сеть уже в стандартной форме IP-пакета. Таким образом, технология MPLS остается прозрачной для остальных IP-сетей.

Обычно в MPLS-сетях используется усовершенствованный по сравнению с описанным алгоритм обработки пакетов. Усовершенствование заключается в том, что удаление метки выполняет не последнее на пути устройство, а *предпоследнее*. Действительно, после того как предпоследнее устройство определит на основе значения метки следующий хоп, метка в MPLS-кадре уже не нужна, так как последнее устройство, то есть выходное устройство LER, должно продвигать пакет на основе значения IP-адреса. Это небольшое изменение

алгоритма продвижения кадра позволяет сэкономить одну операцию над MPLS-кадром. В противном случае последнее вдоль пути устройство должно было бы удалить метку, а уже затем выполнить просмотр таблицы IP-маршрутизации. Эта техника получила название техники **удаления метки на предпоследнем хопе** (Penultimate Hop Popping, **PHP**).

Заголовок MPLS и технологии канального уровня

Заголовок MPLS состоит из нескольких полей (рис. 20.5):

- *Метка* (20 бит). Используется для выбора соответствующего пути коммутации по меткам.
- *Время жизни* (TTL). Данное поле, занимающее 8 бит, дублирует аналогичное поле IP-пакета. Это необходимо для того, чтобы устройства LSR могли отбрасывать «заблудившиеся» пакеты только на основании информации, содержащейся в заголовке MPLS, не обращаясь к заголовку IP.
- *Класс услуги* (Class of Service, CoS). Поле CoS, занимающее 3 бита, первоначально было зарезервировано для развития технологии, но в последнее время используется в основном для указания класса трафика, требующего определенного уровня QoS.
- *Признак дна стека меток*. Этот признак (S) занимает 1 бит.

Концепцию стека меток мы изучим в следующем разделе, а пока для пояснения механизма взаимодействия MPLS с технологиями канального уровня рассмотрим ситуацию, когда заголовок MPLS включает только одну метку. В кадрах канального уровня заголовок MPLS помещается между оригинальным заголовком и заголовком пакета 3-го уровня. На рис. 20.5 этот способ размещения метки показан для кадров PPP и Ethernet. Стандарты MPLS определяют также способ размещения метки в кадрах Frame Relay и ячейках ATM. Поскольку заголовок MPLS помещается между заголовком канального уровня и заголовком IP, его называют **заголовком-вставкой** (shim header).

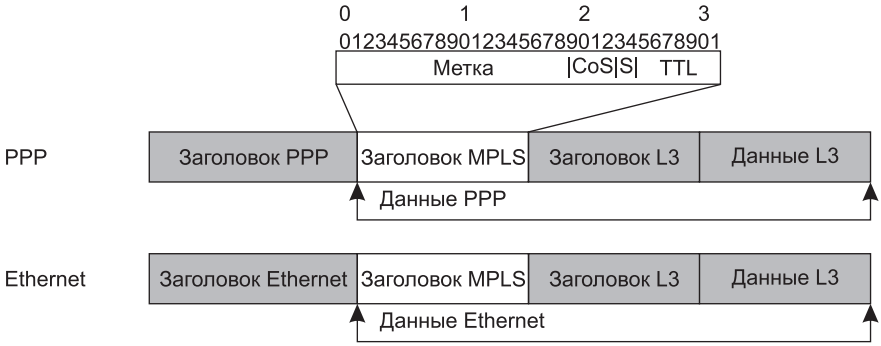


Рис. 20.5. Форматы заголовков нескольких разновидностей технологии MPLS

Продвижение кадра в MPLS-сети происходит на основе метки MPLS и техники LSP, а не на основе адресной информации и техники той технологии, формат кадра которой MPLS использует. Таким образом, если в MPLS применяется кадр Ethernet, то MAC-адреса источника и приемника, хотя и присутствуют в соответствующих полях кадра Ethernet, для продвижения кадров в соединениях Ethernet с двухточечной топологией не используются.

Исключение — случай, когда между двумя соседними устройствами LSR находится сеть коммутаторов Ethernet: MAC-адрес назначения MPLS-кадра потребуется для того, чтобы кадр дошел до следующего устройства LSR, а уже оно будет продвигать его на основании метки. Нахождение MAC-адреса следующего LSR будет в этом случае выполнено стандартным способом с помощью протокола ARP по IP-адресу LSR. Далее для определенности при рассмотрении примеров подразумевается использование формата кадров MPLS/PPP.

Стек меток

Наличие **стека меток** является одним из оригинальных свойств MPLS. Стек меток позволяет создавать систему агрегированных путей LSP с любым количеством уровней иерархии. Для поддержания этой функции MPLS-кадр, который перемещается вдоль иерархически организованного пути, должен включать столько заголовков MPLS, сколько уровней иерархии имеет путь. Напомним, что заголовок MPLS каждого уровня имеет собственный набор полей: метка, CoS, TTL и S. Последовательность заголовков организована как стек, так что всегда имеется метка, находящаяся на вершине стека, и метка, находящаяся на дне стека, при этом последняя сопровождается признаком S = 1. Над метками выполняются следующие операции, задаваемые в поле действий таблицы продвижения:

- *Push* — поместить метку в стек. В случае пустого стека эта операция означает простое присвоение метки пакету. Если же в стеке уже имеются метки, то в результате этой операции новая метка сдвигает старые вглубь стека, сама оказываясь на вершине.
- *Swap* — заменить текущую метку новой.
- *Pop* — выталкивание (удаление) верхней метки, в результате чего все остальные метки стека поднимаются на один уровень.

Продвижение MPLS-кадра всегда происходит на основе метки, находящейся в данный момент на вершине стека. Иерархия меток чаще всего находит свое применение в сетях, разделенных на несколько доменов. Внутри домена продвижение пакетов происходит на основе меток одного из уровней стека, а между доменами — на основе меток другого уровня. Такой подход позволяет независимо организовать внутридоменную и междоменную маршрутизацию пакетов, что во многих случаях оказывается полезным. Здесь можно провести аналогию с использованием MAC-адресов для передачи пакетов внутри IP-подсети и IP-адресов для передачи пакетов между IP-подсетями. Стек меток также оказывается полезным при организации сервиса VPN (соответствующие примеры см. в главе 21).

Рассмотрим работу двух уровней иерархии меток на примере сети, изображенной на рис. 20.6. Сеть состоит из трех MPLS-доменов. На рисунке показаны путь LSP1 в домене 1 и путь LSP2 в домене 2. LSP1 соединяет устройства LER1 и LER2, проходя через устройства LSR1, LSR2 и LSR3. Пусть начальной меткой пути LSP1 является метка 256, которая была присвоена пакету пограничным устройством LER1.

На основании этой метки пакет поступает на устройство LSR1, которое по своей таблице продвижения определяет новое значение метки пакета (272) и переправляет его на вход LSR2. Устройство LSR2, действуя аналогично, присваивает пакету новое значение метки (132) и передает его на вход LSR3. Устройство LSR3, будучи предпоследним устройством в пути LSP1, выполняет операцию *Pop* и удаляет метку из стека. Устройство LER2 продвигает пакет уже на основании IP-адреса. На рисунке также показан путь LSP2 в домене 2. Он соединяет устройства LER3 и LER4, проходя через устройства LSR4, LSR5

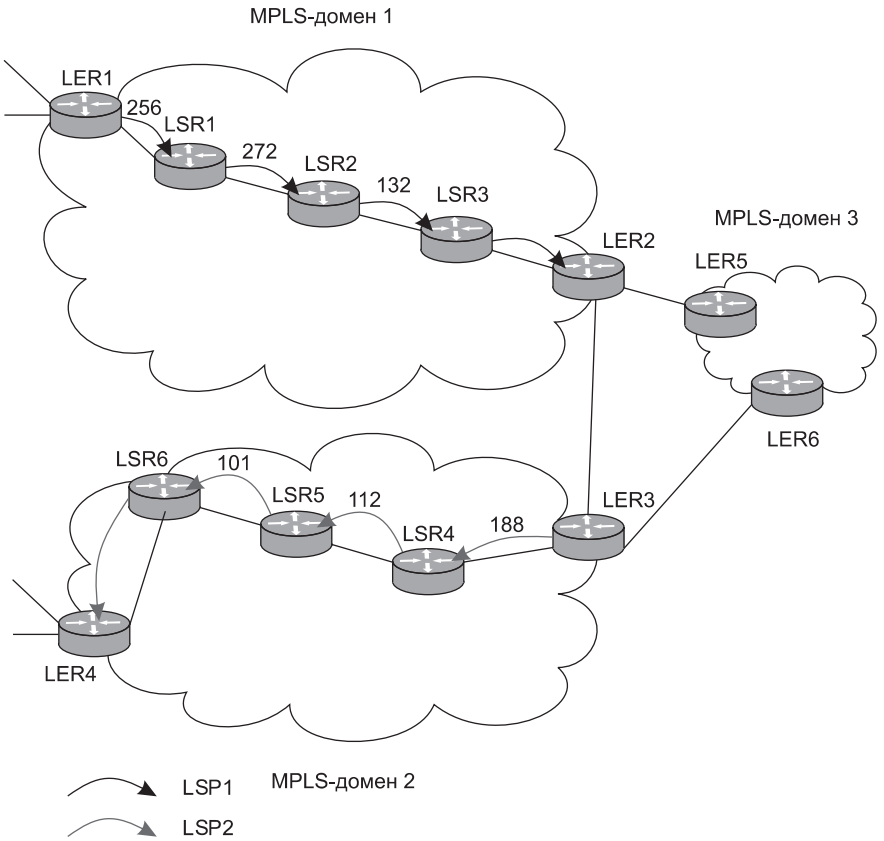


Рис. 20.6. Пути LSP1 и LSP2, проложенные в доменах 1 и 2 MPLS-сети

и LSR6, и определяется последовательностью меток 188, 112, 101. Для того чтобы IP-пакеты могли передаваться на основе техники MPLS не только внутри каждого домена, но и между доменами (например, между устройствами LER1 и LER4), существуют два принципиально разных решения.

- Первое решение состоит в том, что между LER1 и LER4 устанавливается один *одноуровневый* путь коммутации по меткам, соединяющий пути LSP1 и LSP2 (которые в этом случае становятся одним путем). Это простое на первый взгляд решение, называемое **сшиванием** путей LSP, плохо работает в том случае, когда MPLS-домены принадлежат разным поставщикам услуг, не позволяя им действовать независимо друг от друга, так как путь должен быть установлен «из конца в конец» одним из сигнальных протоколов (мы их рассмотрим далее, сейчас детали их работы нам не важны).
- Вторым, более перспективным решением является применение *многоуровневого* подхода к соединению двух MPLS-доменов, принадлежащих, возможно, разным поставщикам услуг.

Для реализации второго подхода в нашем примере нужно создать путь коммутации по меткам второго уровня (LSP3), соединяющий устройства LER1 и LER4. Этот путь определяет

последовательность хопов *между доменами*, а не между внутренними устройствами LSR каждого домена. Так, LSP3 состоит из хопов LER1-LER2-LER3-LER4. В этом отношении многоуровневый подход MPLS концептуально близок подходу протокола BGP, определяющего путь между автономными системами.

Рассмотрим более детально, как работает технология MPLS в случае путей коммутации по меткам двух уровней (рис. 20.7).

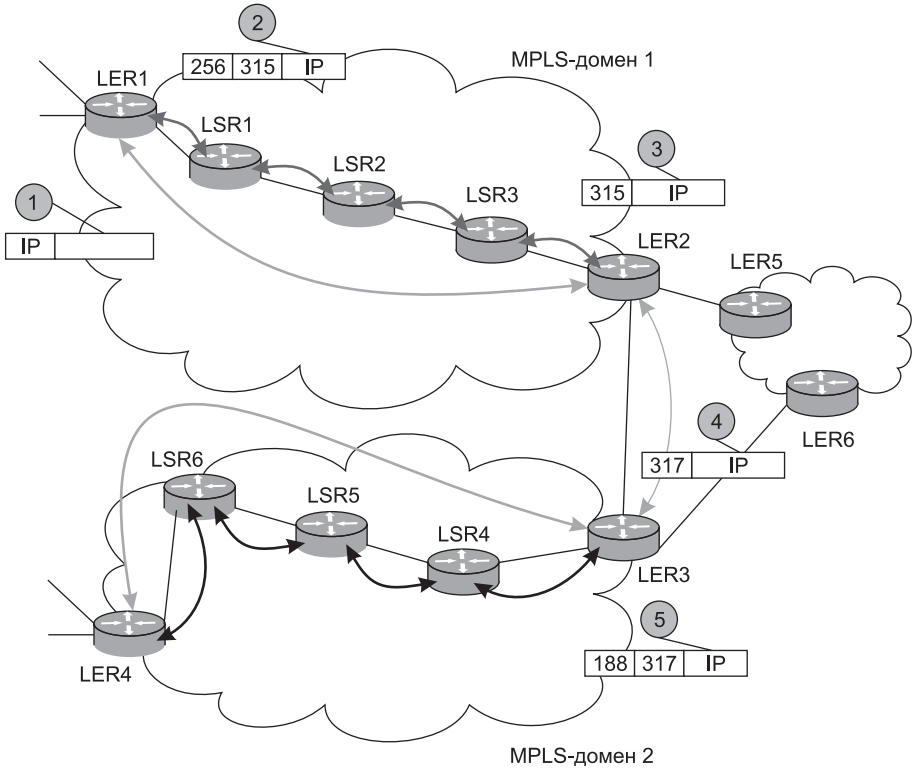


Рис. 20.7. Использование стека меток в иерархии путей

В устройстве LER1 начинаются два пути -- LSP1 и LSP3 (последний показан на рисунке серым цветом), что обеспечивается соответствующей записью в таблице продвижения устройства LER1 (табл. 20.3).

Таблица 20.3. Запись в таблице продвижения LER1

Входной интерфейс	Метка	Следующий хоп	Действия	Выходная метка
...	...	...	...	
S0	—	S1	Push Push	315 256
...	...	...	...	

IP-пакеты, поступающие на интерфейс S0 устройства LER1, продвигаются на его выходной интерфейс S1, где для них создается заголовок MPLS, включающий метку 315 верхнего уровня (LSP3), которая на этот момент является верхушкой стека меток. Затем эта метка проталкивается на дно стека (операция *Push*), а верхней становится метка 256, относящаяся к LSP1. Далее MPLS-кадр с меткой 256 поступает на выходной интерфейс S1 пограничного устройства LER1 и передается на вход LSR1. Устройство LSR1 обрабатывает кадр в соответствии со своей таблицей продвижения (табл. 20.4). Метка 256, находящаяся на вершине стека, заменяется меткой 272 (метка 315, находящаяся ниже в стеке, устройством LSR1 игнорируется).

Таблица 20.4. Запись в таблице продвижения LSR1

Входной интерфейс	Метка	Следующий хоп	Действия	Выходная метка
...	...	...	...	
S0	256	S1	Swap	272
...	...	...	...	

Аналогичные действия выполняет устройство LSR2, которое заменяет метку меткой 132 и отправляет кадр следующему по пути устройству LSR3 (табл. 20.5).

Таблица 20.5. Запись в таблице продвижения LSR3

Входной интерфейс	Метка	Следующий хоп	Действия	Выходная метка
...	...	...	...	
S0	132	S1	Pop	315
...	...	...	...	

Работа устройства LSR3 несколько отличается от работы устройств LSR1 и LSR2, так как оно является *предпоследним* устройством LSR для пути LSP1. В соответствии с записью в табл. 20.5 устройство LSR3 выполняет выталкивание (*Pop*) из стека метки 132, относящейся к пути LSP1, выполняя операцию PHP. В результате верхней меткой стека становится метка 315, принадлежащая пути LSP3. Устройство LER2 продвигает поступивший на его входной интерфейс S0 кадр на основе своей записи таблицы продвижения (табл. 20.6). Устройство LER3 сначала заменяет метку 315 пути LSP3 значением 317, затем проталкивает ее на дно стека и помещает на вершину стека метку 188, которая является меткой пути LSP2, внутреннего для домена 2. Перемещение кадра вдоль пути LSP2 происходит аналогичным образом.

Таблица 20.6. Запись в таблице продвижения LER3

Входной интерфейс	Метка	Следующий хоп	Действия	Выходная метка
...	...	...	...	
S0	315	S1	Swap Push	317 188
...	...	...	...	

Подчеркнем: значение метки междоменного пути LSP3 на границе между доменами не зависит от значений меток, используемых для внутридоменных путей LSP1 и LSP2. Это позволяет операторам доменов изменять значения меток внутридоменных путей независимо друг от друга, например, прокладывая внутридоменные пути по другим маршрутам (а это неизбежно приведет к переназначению меток в каждом из устройств LSR и LER). Важно, что при этом значение междоменной метки при передаче пакета между устройствами LER доменов не меняется, поэтому пакет правильно обрабатывается принимающим устройством LER. Например, LER3 получит пакет от LSR6 со значением метки 317 независимо от того, какое значение имела метка внутридоменного пути LSP1. При сшивании одноуровневых устройств LSP такой независимости доменов добиться нельзя. Описанная модель двухуровневого пути легко может быть расширена для любого количества уровней.

Протокол LDP

Протокол распределения меток (Label Distribution Protocol, **LDP**) позволяет автоматически создавать в сети пути LSP в соответствии с *существующими* в таблицах маршрутизации записями о маршрутах в IP-сети. Протокол LDP является сигнальным протоколом сетей MPLS, принимая во внимание только те записи таблицы маршрутизации, которые созданы с помощью внутренних протоколов маршрутизации, то есть протоколов типа IGP, поэтому режим автоматического создания LSP с помощью протокола LDP иногда называют режимом MPLS IGP (в отличие от режима MPLS TE, когда маршруты выбираются из соображений инжиниринга трафика и не совпадают с маршрутами, выбранными внутренними протоколами маршрутизации). Спецификация LDP дана в документе RFC 5036.

Рассмотрим работу протокола LDP на примере сети, изображенной на рис. 20.8.

Сеть	Следующий хоп
132.100.0.0	LSR2
105.0.0.0	LSR4
192.201.103.0	LSR4

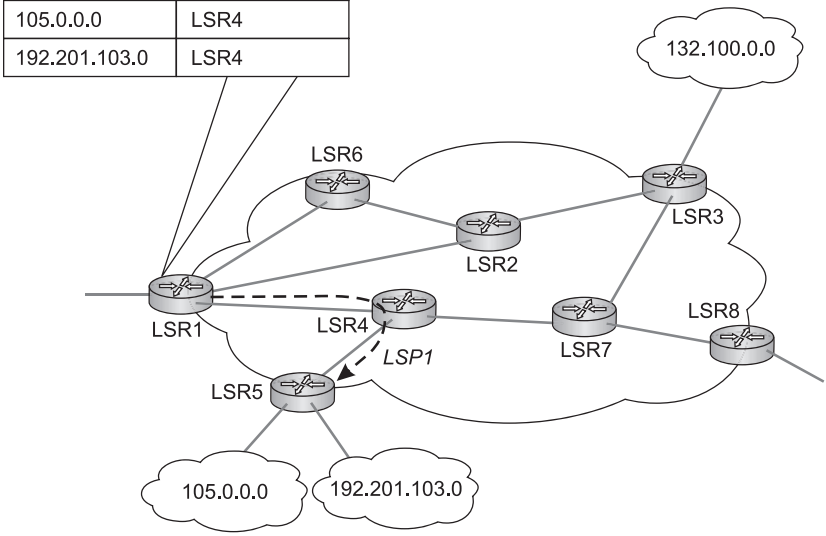


Рис. 20.8. MPLS-сеть с устройствами LSR, поддерживающими LDP

Все устройства LSR сети поддерживают сигнальный протокол LDP. От устройства LSR1 в сети уже установлен один путь LSP1 — по этому пути идет трафик к сетям 105.0.0.0 и 192.201.103.0. Это значит, что таблица FTN (отображающая сети назначения на устройства LSP) у LSR1 соответствует табл. 20.7.

Таблица 20.7. Таблица FTN устройства LSR1

Признаки FEC	Метка
105.0.0.0; 192.201.103.0	231

Метка 231 в этой таблице соответствует пути LSP1. Рассмотрим функционирование протокола LDP в ситуации, когда в результате работы протоколов маршрутизации или же после ручной модификации администратором сети в таблице маршрутизации устройства LSR1 появилась запись о новой сети назначения, для которой в сети поставщика услуг еще не проложен путь коммутации по меткам. В нашем случае это сеть 132.100.0.0 (она закрашена в таблице маршрутизации LSR1) и для нее нет записи в таблице FTN. В этом случае устройство LSR1 автоматически инициирует процедуру прокладки нового пути. Для этого оно запрашивает по протоколу LDP метку для новой сети 132.100.0.0 у маршрутизатора, IP-адрес которого в таблице маршрутизации указан для данной сети как адрес следующего хопа.

Но для того чтобы воспользоваться протоколом LDP, нужно сначала установить между устройствами LSR сеанс LDP, так как этот протокол работает в режиме установления соединений. Сеансы LDP устанавливаются между соседними маршрутизаторами автоматически. Для этого каждое устройство LSR, на котором развернут протокол LDP, начинает посылать своим соседям сообщения HELLO. Эти сообщения посылаются по групповому IP-адресу 224.0.0.2, который является адресом всех маршрутизаторов подсети. Если соседний маршрутизатор также поддерживает протокол LDP, то он в ответ устанавливает сеанс TCP через порт 646 (этот порт закреплен за протоколом LDP). В результате обмена сообщениями HELLO все поддерживающие протокол LDP устройства LSR обнаруживают своих соседей и устанавливают с ними сеансы, как показано на рис. 20.9 (для простоты на рисунке представлены не все сеансы LDP, существующие в сети).

Будем считать, что между устройствами LSR1 и LSR2 установлен сеанс LDP. Тогда при обнаружении новой записи в таблице маршрутизации, указывающей на устройство LSR2 в качестве следующего хопа, устройство LSR1 просит устройство LSR2 назначить метку для нового пути к сети 132.100.0.0. Говорят, что устройство LSR2 находится ниже по потоку (downstream) относительно устройства LSR1 на пути к сети 132.100.0.0. Соответственно, устройство LSR1 расположено выше по потоку для устройства LSR2 относительно сети 132.100.0.0. Естественно, что для других сетей назначения у устройства LSR1 имеются другие соседи вниз по потоку, а у устройства LSR2 — другие соседи вверх по потоку.

Причина, по которой значение метки для нового пути выбирается соседом ниже по потоку, понятна — эта метка, которая имеет локальное значение на двухточечном соединении между соседними устройствами, будет использоваться именно этим устройством для того, чтобы понимать, к какому пути LSP относится пришедший MPLS-кадр. Поэтому устройство ниже по потоку выбирает уникальное значение метки, исходя из неиспользованных значений меток для своего интерфейса, который связывает его с соседом выше по потоку. Для получения значения метки устройство LSR1 выполняет несложный по формату запрос метки протокола LDP (рис. 20.10).

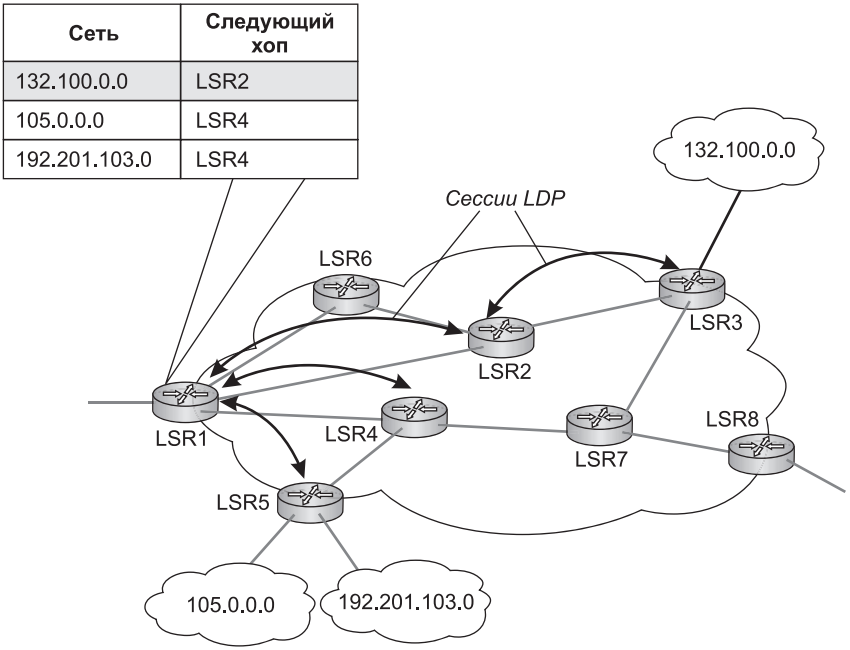


Рис. 20.9. Сеансы LDP устанавливаются между непосредственными соседями

Запрос метки (0×0401)	Длина сообщения
Идентификатор сообщения	
Элемент FEC	

Рис . 20 .10 . Формат LDP-запроса метки

Идентификатор сообщения требуется для того, чтобы при получении ответа можно было однозначно сопоставить ответ некоторому запросу (устройство может послать несколько запросов до получения ответов на каждый из них). В нашем примере в качестве элемента FEC указан адрес 132.100.0.0. Устройство LSR2, приняв запрос, находит, что у него также нет проложенного пути к сети 132.100.0.0, и поэтому передает LDP-запрос следующему устройству LSR, адрес которого указан в его таблице маршрутизации в качестве следующего хопа для сети 132.100.0.0. В примере, показанном на рис. 20.9, таким устройством является LSR3, на котором путь коммутации по меткам должен закончиться, так как следующий хоп ведет за пределы MPLS-сети данного оператора.

ПРИМЕЧАНИЕ

Возникает вопрос: как устройство LSR3 узнает о том, что является последним в сети поставщика услуг на пути к сети 132.100.0.0? Дело в том, что сеансы LDP устанавливаются только между устройствами одного поставщика услуг, поэтому отсутствие сеанса LDP со следующим хопом маршрута и говорит устройству LSR, что оно является последним в своем домене для данного пути LSP.

Устройство LSR3, обнаружив, что для пути к сети 132.100.0.0 оно является пограничным, назначает для прокладываемого пути метку, еще не занятую его входным интерфейсом S0, и сообщает об этой метке устройству LSR2 в LDP-сообщении, формат которого представлен на рис. 20.11. Пусть это будет метка 231.

Отображение метки (0×0400)	Длина сообщения
Идентификатор сообщения	
Элемент FEC	
Метка	

Рис . 20 .11 . Формат отображения метки на элемент FEC протокола LDP

В свою очередь, устройство LSR2 назначает неиспользуемую его интерфейсом S0 метку и сообщает об этом в LDP-сообщении отображения метки устройству LSR1. После этого новый путь коммутации по меткам от LSR1 к сети 132.100.0.0 считается проложенным (рис. 20.12) — вдоль него пакеты начинают передаваться уже на основе меток и таблиц продвижения, а не IP-адресов и таблиц маршрутизации.

Сеть	Следующий хоп
132.100.0.0	LSR2
194.15.17.0	LSR2
201.25.10.0	LSR2
105.0.0.0	LSR4
192.201.103.0	LSR4

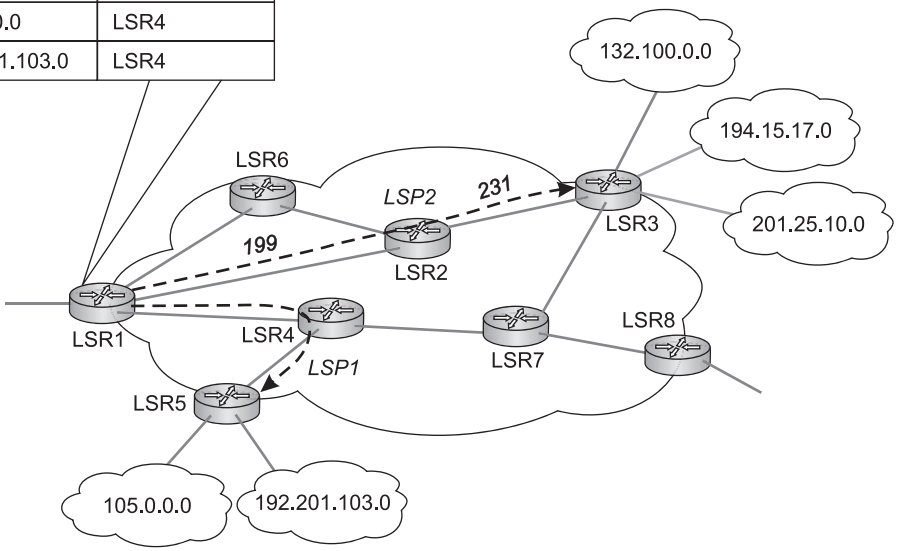


Рис. 20.12. Новый путь LSP2

Было бы нерационально прокладывать отдельный путь для каждой сети назначения каждого маршрутизатора, и устройства LSR стараются строить агрегированные пути коммутации по меткам и передавать вдоль них пакеты, следующие к некоторому набору сетей. Так, на рис. 20.12 устройство LSR1 передает по пути LSP2 пакеты, следующие не только к сети 132.100.0.0, но и к сетям 194.15.17.0 и 201.25.10.0, информация о которых появилась уже после того, как путь LSP2 был проложен.

Мы рассмотрели только один режим работы протокола LDP, носящий длинное название «Упорядоченный режим управления распределением меток с запросом устройства вниз по потоку». Здесь под *упорядоченным* понимается такой режим, когда некоторое промежуточное устройство LSR не передает метку для нового пути устройству LSR, лежащему выше по потоку, до тех пор пока не получит метку для этого пути от устройства LSR, лежащего ниже по потоку. В нашем случае устройство LSR2 ждало получения метки от LSR3 и уже потом передало метку устройству LSR1. Но существует и другой режим управления распределением меток, который называется *независимым*. При независимом управлении распределением меток LSR может назначить и передать метку, не дожидаясь прихода сообщения от своего соседа, лежащего ниже по потоку. Например, устройство LSR2 могло бы назначить и передать метку 199 устройству LSR1, не дожидаясь прихода метки 231 от устройства LSR3. Поскольку метки имеют локальное значение, результат изменения режима сохранился бы.

Существуют также два метода распределения меток — распределение по запросу от лежащего ниже по потоку устройства и распределение без запроса. Для нашего случая это значит, что если бы устройство LSR2 обнаружило в своей таблице маршрутизации запись о новой сети 132.100.0.0, оно могло бы назначить метку новому пути и передать ее устройству LSR1 без запроса. Поскольку при этом устройство LSR2 не знает своего соседа выше по потоку (таблица маршрутизации не говорит об этом), оно передает эту информацию всем своим соседям по сеансам LDP. В этом варианте работы протокола LDP устройства LSR могут получать альтернативные метки для пути к некоторой сети; выбор наилучшего пути осуществляется обычным для IP-маршрутизатров (которыми устройства LSR являются по совместительству) способом — на основании наилучшей метрики, выбираемой протоколом маршрутизации. Как видно из описания, существует два независимых параметра, определяющих вариант работы протокола LDP: режим управления распределением меток и метод распределения меток. Каждый параметр имеет два значения — значит, всего существует четыре режима работы протокола LDP. Протокол LDP чаще всего функционирует в режиме независимого управления распределением меток без запроса. Упорядоченное управление распределением меток требуется при прокладке путей LSP, необходимых для инжиниринга трафика.

Инжиниринг трафика в MPLS

Технология MPLS поддерживает технику инжиниринга трафика, описанную в главе 5. Напомним, что под инжинирингом трафика понимается нахождение путей прохождения трафика через сеть, отличных от кратчайших маршрутов с минимальным суммарным количеством хопов между узлами сети. Задача инжиниринга состоит в нахождении путей, удовлетворяющих определенным условиям — например, проходящим через линии с максимальной свободной, незарезервированной для определенных целей пропускной способностью, или же не проходящие через определенную группу линий связи.

Существуют два основных варианта технологии MPLS TE, поддерживающей инжиниринг трафика:

- на основе протокола RSVP;
- на основе сегментной маршрутизации SR-MPLS (Segment Routing MPLS).

Оба эти варианта используют нахождение пути на графе сети протоколом маршрутизации состояния связей — OSPF или IS-IS. Для решения задачи инжиниринга трафика необходимы специальные версии этих протоколов, называемые версиями OSPF TE или IS-IS TE, то есть версиями с TE-расширениями. Эти версии делают объявления о состоянии связи с дополнительными параметрами, необходимыми для принятия решения об инжиниринге маршрута. В дополнение к обычной метрике — количеству хопов — в состояние связи могут включаться следующие метрики:

- максимальная пропускная способность линии;
- зарезервированная пропускная способность линии;
- свободная (оставшаяся, незарезервированная пропускная способность) линии;
- «цвет» линии — принадлежность к одной из групп, задаваемых администратором;
- величина задержки линии.

В результате обмена объявлениями состояния связей каждый маршрутизатор знает точный граф сети с ребрами, помеченными (нагруженными) одной или несколькими такими дополнительными метриками. Эти метрики являются ограничениями при поиске кратчайшего пути на графе. Результатом работы протокола с расширением TE является граф с ребрами, нагруженными дополнительными параметрами. Такой граф составляет **базу данных инжиниринга трафика** (Traffic Engineering Database, **TED**).

Для нахождения TE-пути (называемого также TE-туннелем) используется специальный алгоритм поиска кратчайшего пути на графе. На первом этапе этот алгоритм просто отбрасывает линии связи, которые не удовлетворяют заданному администратором сети ограничениям, — например, отбрасываются линии связи, у которых максимальная пропускная способность меньше заданного порогового значения, или же задержки, вносимые линией, слишком велики. А затем работает стандартный алгоритм нахождения кратчайшего пути на графе (алгоритм Дейкстры), но уже на графе, у которого имеются только линии связи, удовлетворяющие заданным ограничениям. Такой алгоритм нахождения кратчайшего пути называется **алгоритмом нахождения кратчайшего пути с ограничениями** (constrains), **CSPF**.

Алгоритм CSPF не должен работать на каждом маршрутизаторе сети. Достаточно его применить на одном из узлов сети. Обычно этим узлом является пограничный узел LER, который находит TE-путь с помощью алгоритма CSPF, а затем устанавливает найденный путь в каждом промежуточном узле сети. Возможен вариант, когда для нахождения TE-пути используется внешний по отношению к сети узел, называемый узлом PCE (Path Computation Element). Этот внешний узел должен получать маршрутные объявления протокола OSPF TE (или протокола IS-IS TE).

Теперь рассмотрим оба варианта технологии MPLS TE, которые используют описанную процедуру нахождения TE-пути и устанавливают его в сети MPLS. Эти варианты отличаются именно способом установки TE-пути.

Вариант RSVP основан на модифицированной версии протокола RSVP технологии IntServ, кратко описанной в главе 14. Эта версия под названием RSVP-TE выполняет две функции:

- устанавливает в сети маршрутизаторов TE-путь, найденный алгоритмом CSPF,
- резервирует пропускную способность на каждом из маршрутизаторов вдоль устанавливаемого TE-пути.

Маршрутизаторы в сети RSVP-TE являются стандартными MPLS узлами, то есть узлами LSR и LER. Они продвигают пакеты на основе MPLS-меток и соответствующих действий в их таблицах продвижения LFIB. Алгоритм CSFP, работающий на входном маршрутизаторе LER или на внешнем сервере PCE, описывает TE-путь как последовательность узлов сети. Сообщение протокола RSVP-TE вызывает на каждом маршрутизаторе генерацию метки, имеющую локальное значение, определяющее для данного маршрутизатора данный TE-путь (используется упорядоченное управление распределением меток). Одновременно с генерацией метки маршрутизатор резервирует для нового TE-пути пропускную способность — тем же способом, которым это делает протокол RSVP технологии IntServ в IP-сетях. Маршрутизатор уменьшает значение своей свободной пропускной способности на величину резервации и в последующих маршрутных объявлениях OSPF TE (или IS-IS TE) указывает уже измененное значение, тем самым обеспечивая *динамический* характер отслеживания свободной пропускной способности каждой линии сети и тем самым обеспечивая гарантии пропускной способности для отдельных TE-путей (LSP-путей).

Применение подхода, основанного на применении протокола RSVP-TE, имеет те же недостатки, что и применение протокола RSVP в технологии IntServ для IP-сетей, — в первую очередь необходимость для каждого маршрутизатора сети хранить информацию о состоянии резервирования пропускной способности. Из-за этого недостатка вариант с применением протокола RSVP-TE не получил развития и не считается перспективным.

Сегментная маршрутизация SR-MPLS не требует хранения в маршрутизаторах сети информации о состоянии TE-пути, поэтому она гораздо более масштабируема и считается на сегодняшний день основной технологией инжиниринга трафика в MPLS-сетях. Кроме того, идея сегментной маршрутизации реализована не только для сетей MPLS, но и для сетей IPv6, что делает ее в значительной степени универсальной.

Сегментная маршрутизация представляет собой один из вариантов *маршрутизации от источника*, когда маршрут в виде последовательности узлов сети хранится не в таблицах маршрутизации узлов, а в заголовке каждого пакета. Не случайно в качестве конкретных реализаций идеи сегментной маршрутизации выбраны технологии MPLS и IPv6 — в этих технологиях заголовки пакетов позволяют переносить информацию о последовательности узлов маршрута. В MPLS эта возможность поддерживается стеком меток, а в IPv6 — заголовком нового типа «Сегментная маршрутизация» (Segment Routing Header, SRH).

Общая архитектура сегментной маршрутизации (общая для MPLS и IPv6) определена в RFC 8402.

Эта спецификация определяет основные понятия и термины этой технологии, в частности, определяет сегменты маршрута.

Сегменты маршрута — это последовательность *инструкций*, определяющих пошаговую обработку пакета при его продвижении по TE-пути. Наиболее распространенный вид инструкции имеет топологическое содержание, например передачу пакета узлу N или на интерфейс ge-0/1/1. Но возможен и любой другой вид инструкции, задающий определенные действия над пакетом, например компрессирование его заголовка. То есть понятие инструкции обобщает действия, выполняемые над пакетом в ходе его перемещения по

заданному маршруту. Интуитивное понимание сегментной маршрутизации как задания последовательности сегментов маршрута имеет право на существование, но не отражает точку зрения авторов этой технологии, которые имели в виду небольшие сегменты программного кода, являющиеся строительными блоками («кусочками») программы.

Последовательность сегментов, задающая последовательность обработки пакета при его прохождении через сеть, называется SR-политикой (SR-Policy) или **SR-путем**. SR-путь может задавать не только TE-путь, но и обычный SPF-путь, то есть кратчайший путь, найденный на основе стандартной метрики количества хопов. Кроме того, одни части общего пути могут быть TE-частями, а другие — SFP-частями. В этом проявляется гибкость технологии SR-MPLS.

В нашем кратком описании технологии SR-MPLS мы сосредоточимся только на топологических инструкциях.

В топологических инструкциях используются идентификаторы узлов и интерфейсов узлов. Эти идентификаторы имеют формат метки MPLS (то есть 20 бит) и используются в качестве таковой в стеке меток заголовка пакета MPLS. Узлы сети SR-MPLS продвигают пакеты *стандартным* для узлов MPLS способом, то есть находя для верхней метки стека запись в таблице продвижения LFIB и выполняя действия, задаваемые этой записью, — например, передачу пакета следующему хопу и замену метки на новую (действие *Swap*). Идентификаторы узлов и интерфейсов являются частным случаем идентификаторов сегментов, но мы ограничимся рассмотрением идентификаторов только этих двух типов.

Идентификатор узла Node-SID представляет собой *глобальный* идентификатор. Его глобальность означает, что он уникален в пределах данной сети и, следовательно, однозначно определяет узел сети. В этом его принципиальное отличие от обычных меток MPLS, которые имеют локальное значение для узла и уникальны только в пределах одного интерфейса этого узла. Технология SR-MPLS является исключением из общего для техники виртуальных каналов правила. О таких исключениях мы говорили в начале нашего изучения технологии MPLS.

Идентификатор Node-SID присваивается узлу администратором сети из специального диапазона значений меток, называемого *глобальным блоком меток* (Segment Routing Global Block, SRGB). Каждый узел сети знает этот диапазон (его значение также конфигурируется администратором сети на каждом узле), поэтому метки из него обрабатываются узлами в соответствии с особыми правилами, то есть в соответствии с семантикой сегментной маршрутизации.

Идентификатор интерфейса Adj-SID является локальным для каждого узла (и его значение уникально для всего узла, то есть в пределах его всех интерфейсов). Идентификатор Adj-SID (Adjecency SID) описывает связь (присоединение) данного узла с другими узлами. Его значение определяется самим узлом, который назначает его из своего *локального блока меток* (Segment Routing Local Block, SRLB), который также конфигурируется на каждом узле сети администратором.

На рис. 20.13 представлена сеть SR-MPLS, состоящая из 5 узлов, и показаны значения идентификаторов Node-SID всех узлов, а также значения идентификаторов Adj-SID для некоторых интерфейсов. Из рисунка видно, что глобальные метки взяты из диапазона 16 000–23 999, а локальные — из диапазона 24 000–31 999 (это рекомендуемые, но не обязательные диапазоны). Каждый узел также идентифицируется своим петлевым IP-адресом (loopback address).

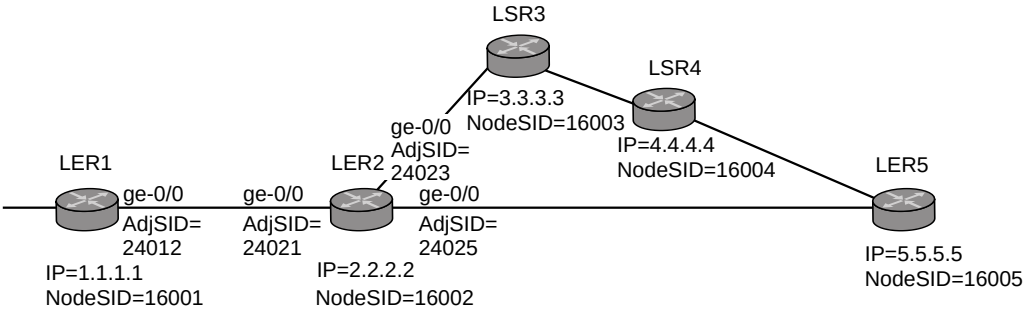


Рис. 20.13. Пример домена SR-MPLS

Для определенности будем считать, что в сети работает протокол маршрутизации OSPF. Для того чтобы технология SR-MPLS могла работать в режиме TE, необходима поддержка двух расширений этого протокола — расширения TE (RFC 3630 и RFC 5329) и расширения SR-MPLS (RFC 8665–8667). Расширения SR-MPLS нужны для того, чтобы каждый узел сети объявлял о себе и о своих связях с соседями. Например, узел LER1 посылает в сеть объявление о себе (помимо стандартных объявлений о связях и объявлений TE о значениях ограничительных параметров линий связи):

```
{Advertising Router: 1.1.1.1, SID=16001}
```

и о своей связи с узлом LSR2:

```
{Advertising Router: 1.1.1.1, Adj-SID = 24012, Neighbor: 2.2.2.2, Interface = ge0/0}
```

На основе этих объявлений каждый узел формирует записи в своей таблице продвижения LFIB обо всех узлах, поддерживающих SR-MPLS и их связях. Эти записи являются результатом работы алгоритма поиска кратчайшего пути SPF на каждом узле — как обычно, этот алгоритм находит кратчайшие маршруты от себя до всех остальных узлов сети и помещает следующий хоп для каждого адреса назначения в таблицу LFIB.

Нужно подчеркнуть, что на каждом узле работает алгоритм SPF, а не CSPF, поэтому найденные маршруты не являются маршрутами TE. Это важное свойство технологии SR-MPLS — *она может работать в обычном режиме IP-маршрутизации*, передавая пакеты по стандартным кратчайшим маршрутам, но только на основе меток Node-SID — таким же образом, как это делает сеть MPLS с протоколом LDP с обычными (локальными) метками. Каким образом сеть SR-MPLS обеспечивает передачу трафика по TE-путям, мы увидим немного позже, а пока посмотрим на таблицу продвижения узла LSR2 (точнее, на ее наиболее значимые для нас поля).

Таблица 20.8. Записи в таблице продвижения LSR2

Метка	Следующий хоп	Действия	Выходная метка
16001	1.1.1.1 (LER1)	Swap	16001
16002	-	Pop	Следующая из стека
16003	3.3.3.3	Swap	16003
16004	3.3.3.3	Swap	16004

Метка	Следующий хоп	Действия	Выходная метка
16005	5.5.5.5	Swap	16005
24021	1.1.1.1	Pop	Следующая из стека
24023	3.3.3.3	Pop	Следующая из стека
24025	5.5.5.5	Pop	Следующая из стека

Мы видим, что метки Node-SID при прохождении пакетов через узел LSR2 не меняют своего значения. Такая обработка меток этого типа (с помощью операции *Swap* на такую же метку) является следствием *глобального* характера этих меток — они играют роль постоянного адреса назначения пакета, роль, аналогичную роли IP-адреса назначения, который также не меняется при прохождении IP-пакета через сеть маршрутизаторов. Можно сказать, что активная глобальная метка (то есть находящаяся на вершине стека меток) дублирует IP-адрес назначения пакета и, если сравнить таблицу LFIB узла LSR2 с таблицей IP-маршрутизации этого узла, то мы увидим записи, вызывающие передачу пакетов, но записанные с использованием IP-адресов.

Таким образом, записи таблиц продвижения LFIB для идентификаторов Node-SID продвигают пакеты вдоль путей, выбранных алгоритмом SPF на основе традиционной метрики, такой как количество хопов. Поэтому в том случае, когда стек меток SR-Policy состоит только из меток Node-SID, трафик будет следовать теми же путями, что и в сети MPLS LDP, но без привлечения протокола LDP, его работу выполняет протокол OSPF (который работает и в сети MPLS LDP).

Рассмотрим в качестве примера прохождение пакета, направленного узлу LER5 через сеть. Пусть входной маршрутизатор LER1 получил пакет со следующим состоянием списка меток (SR-Policy):

```
{16001, 16005}.
```

Узел LER1 имеет таблицу LFIB, аналогичную таблице узла LSR2, с записями для всех меток сети, в том числе и такими:

Таблица 20.9. Записи в таблице продвижения LER1

Метка	Следующий хоп	Действия	Выходная метка
16001	-	Pop	Следующая
16005	2.2.2.2	Swap	1605
24012	2.2.2.2	Pop	Следующая

Узел LER1 видит, что активная метка принадлежит ему, удаляет ее из стека, после чего активной становится метка 16005. Запись для этой метки показывает, что пакет нужно передать узлу 2.2.2.2 (LSR2), при этом значение метки изменять не нужно (*Swap* на то же значение). Пакет следует кратчайшему пути SPF до узла назначения LER5, потому что этот путь был вычислен протоколом OSPF в каждом узле сети и установлен в таблицы LFIB.

Получив пакет, узел LSR2 обрабатывает его в соответствии с записью его таблицы LFIB для активной метки, которой является метка 16005 (см. табл. 20.8). Запись указывает на

действие, аналогичное действию, которые выполнил узел LER1, — метка сохраняется. Пакет передается следующему хопу, которым является узел назначения LER5.

В первом примере мы в списке SR-Policy задали путь, совпадающий с путем SPF, — и он был соблюден. Можно задать такой путь и короче, только указав финальную метку, в нашем случае 16005 — результат был бы тот же.

Теперь посмотрим, как задать TE-путь, отличающийся от SPF-пути. Пусть мы хотим послать трафик к узлу LER5 по длинному маршруту, через узлы LSR4. Составить список меток для соответствующего TE-пути можно двумя способами.

Первый — использовать в списке только идентификаторы узлов Node-SID и указать все промежуточные узлы TE-пути: [16001, 16002, 16003, 16004, 16005]. Для топологии нашего примера такой список будет работать. Узел LSR2, удалив собственную метку из стека (действие Pop), делает активной следующую метку 16003, поэтому он передает пакет не узлу LER5, а узлу LSR3, точно следуя инструкциям из списка SR-Policy. Этот способ можно оптимизировать, если заметить, что после посещения узла LSR3 TE-путь к узлу LER5 совпадает с SPF-путем к этому узлу. Поэтому не обязательно указывать в списке SR-Policy метки *всех* промежуточных узлов. Достаточно тех, которые указывают на «сворачивания» с SPF-пути на TE-путь. Поясним эту мысль примером: в списке TE-пути к узлу LER5 можно оставить только промежуточный узел LSR3, те есть сократить список до [16003, 16005], и результат продвижения пакета будет тот же самый.

Второй способ состоит в использовании меток связей Adj-SID в списке SR-Policy для TE-пути. Например, предыдущий список можно представить так: [16002, 24023, 16005]. Вторая метка списка является меткой связи LSR2 с LSR3 через интерфейс ge-0/1, и она вынуждает узел LSR2 передать пакет узлу LSR3, а не LER5. Применение меток Adj-SID в нашем примере не является обязательным, список, состоящий из последовательности меток узлов Node-SID, выполняет ту же работу. Однако существуют топологии, для которых применение меток Adj-SID является единственным способом задать требуемый TE-путь. Представим, что между LSR2 и LSR3 имеются две связи, помеченные метками 24023 и 24123, и мы по какой-то причине хотим задать путь, проходящий через одну из них. Без метки Adj-SID эту задачу решить невозможно, узел LSR2 не будет точно знать, по какой из связей передавать пакет. Задание в списке SR-Policy метки Adj-SID решит эту задачу, список будет выглядеть так: [16002, 24123, 16005].

Обычной практикой является применение меток обоих типов для задания TE-пути.

Теперь рассмотрим варианты архитектуры сети SR-MPLS, отличающиеся друг от друга типом элемента сети, вычисляющего TE-путь и устанавливающего его в первый узел пути, то есть в пограничный маршрутизатор сети.

В первом варианте это делает каждый пограничный маршрутизатор LER. Для этого он собирает топологическую информацию о графе сети, включающую TE-метрики и значения Node-SID и Adj-SID каждого маршрутизатора и каждой связи, используя протокол OSPF или IS-IS с соответствующим расширениями. Затем этот маршрутизатор запускает алгоритм CSPF для нахождения TE-пути и формирует список меток SR-Policy для этого пути, а также формирует запись в таблице FTN, отображающую класс FECn на данный путь. При приходе пакета из класса FECn, пограничный маршрутизатор помещает в его заголовок стек меток SR-Policy и отправляет в путь.

В крупных сетях нагрузка на пограничные маршрутизаторы дополнительной работой по вычислению TE-путей может оказаться слишком большой, мешающей основной ра-

боте маршрутизатора по продвижению пакетов. Поэтому эту работу поручают внешним серверам, которые называются SR-MPLS-контроллерами. SR-MPLS-контроллер может работать на основе различных протоколов, связывающих его с узлами SR-MPLS-сети. Наиболее популярными являются протоколы PCEP (Path Computation Element Protocol) и расширения BGP.

На рис. 20.14 показана сеть SR-MPLS с контроллером на основе протокола PCEP. Каждый маршрутизатор сети SR-MPLS передает контроллеру топологическую информацию по протоколу PCEP, работающему поверх соединений TCP. Для этого каждый маршрутизатор имеет в своем составе специальный программный блок, называемый клиентом PCC (Path Computation Client). Клиент собирает всю топологическую информацию, необходимую для построения базы данных TED, — идентификаторы узлов Node-SID, идентификаторы связей Adj-SID, ШЗ-адреса узлов подсетей, состояние связей, метрики связей, включая TE-метрики. Эту информацию он в компактном виде передает в контроллер, но не периодически, а только при изменении топологической информации, например, изменении состояния связи или метрики лини связи. Контроллер на основании этой информации создает базу топологических данных сети SR-MPLS TED и с помощью алгоритма CSPF находит TE-пути в форме списков SR-Policy. Затем контроллер передает каждый список соответствующему пограничному узлу, который должен использовать этот список для сегментной маршрутизации, с помощью протокола PCEP.

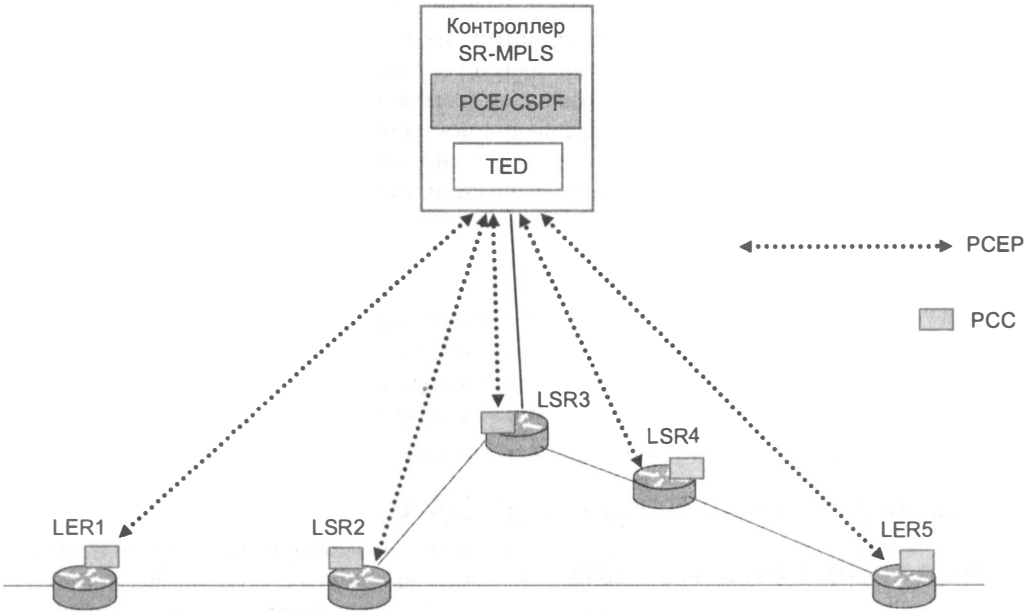


Рис. 20.14. Контроллер SR-MPLS на основе протокола PCEP

Если сравнить диаграмму рис. 20.14 с архитектурой сети на рис. 16.9 «Архитектура сети SDN OpenFlow», то можно увидеть, что они иллюстрируют одну и ту же идею — идею программно-управляемой сети (SDN) с помощью центрального контроллера, собирающего информацию о состоянии узлов и связей в сети, находящего на основании этой информации

путь, удовлетворяющий некоторым критериям, а затем устанавливающий этот путь в узлах сети. Так что сеть SR-MPLS с контроллером PCEP можно считать одной из реализаций подхода SDN. Нетрудно увидеть и разницу между Openflow/P4 SDN и SR-MPLS PCEP SDN — в последнем случае узлы сети не столь кардинально отличаются от стандартных IP/MPLS-маршрутизаторов, как коммутаторы Openflow или P4 отличаются от стандартных Ethernet-коммутаторов. В случае узлов SR-MPLS вся стандартная функциональность маршрутизатора IP/MPLS сохраняется и к ней добавляется функциональность сегментной маршрутизации (все стандарты, регламентирующие эту функциональность, не зря носят название «расширения»). В случае устройств SDN Openflow/P4 функциональность коммутатора Ethernet почти полностью заменяется новой, остается только формат заголовка.

Программирование маршрутов с помощью контроллера PCEP — это недостаточно масштабируемое решение. Контроллер должен поддерживать TCP-соединение с каждым узлом сети для того, чтобы собрать необходимую топологическую информацию для выбора маршрута. Более масштабируемым решением является контроллер на основе расширений протокола BGP.

В этом случае на одном из узлов сети SR-MPLS (возможно, на нескольких) должен работать протокол BGP LS (Link State), который является расширением BGP, предназначенным для передачи данных состояния связей сети в модифицированных сообщениях BGP UPDATE. Протокол BGP LS экспортирует данные состояния связей из базы данных TED, созданной внутренним протоколом маршрутизации — OSPF TE или IS-IS TE.

В контроллере SR-MPLS также работает протокол BGP-LS, с которым агент BGP LS, установленный на узле сети SR-MPLS, устанавливает соединение и передает по нему данные базы TED. Контроллер на основании этих данных вычисляет TE-маршрут в форме списка узлов SR-Policy и передает его для реализации пограничному маршрутизатору сети SR-MPLS для отработки. Есть два протокола, которыми контроллер может воспользоваться для этой цели, — протокол PCE или же недавно стандартизованное расширение протокола BGP, называемое BGP SR (RFC 9830).

Преимуществами применения протокола BGP (для выполнения двух работ — передачи контроллеру информации базы состояния связей TED и передачи пограничному маршрутизатору сети вычисленного маршрута) по сравнению с протоколом PCEP являются: более масштабируемое решение, не требующее связи контроллера со всеми маршрутизаторами сети SR-MPLS, а также возможность помещения контроллера за пределами сети SR-MPLS, так как сессию BGP можно устанавливать и между различными автономными системами.

Отказоустойчивость путей в MPLS

Общая характеристика

MPLS поддерживает несколько механизмов обеспечения отказоустойчивости или, в терминах SDH, механизмов *автоматического защитного переключения* маршрута в случае отказа какого-либо элемента сети: интерфейса LSR, линии связи или LSR в целом. Если путь устанавливается с помощью протокола LDP, то существует единственная возможность защиты пути — его восстановление с помощью распределенного механизма нахождения нового пути средствами протоколов маршрутизации. Это абсолютно тот же механизм, который используется в IP-сетях при отказе линии или маршрутизатора. Время восстанов-

ления пути зависит от применяемого протокола маршрутизации и сложности топологии сети. Обычно это десятки секунд или несколько минут.

В том случае, когда путь является TE-путем, в технологии MPLS разработано несколько механизмов его восстановления. Эти механизмы иллюстрирует рис. 20.15, на котором показан основной путь LSP1, соединяющий устройства LSR1 и LSR8. Будем считать, что путь LSP1 является TE-путем.

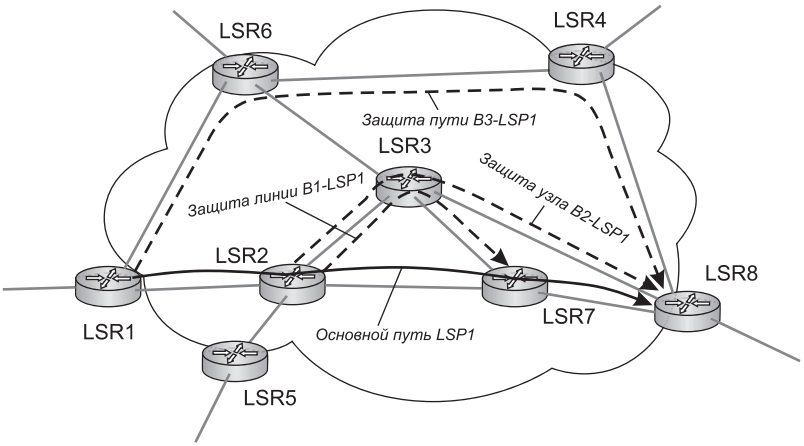


Рис. 20.15. Защитные механизмы MPLS

- *Восстановление пути его начальным узлом* представляет собой традиционное (с помощью протокола маршрутизации) повторное нахождение нового пути, обходящего отказавший элемент сети. Отличие от восстановления пути LDP заключается только в том, что прокладкой нового пути занимается лишь один узел сети, а именно начальный узел пути (в нашем примере — узел LSR1).
- *Защита линии* организуется между двумя устройствами LSR, непосредственно соединенными линией связи. Обходной маршрут находится заранее, до отказа линии, и заранее прокладывается между этими устройствами таким образом, чтобы обойти линию связи в случае ее отказа. В нашем примере такой вариант защиты установлен для линии, соединяющей узлы LSR2 и LSR7. Обходной путь B1-LSP1 проложен через узел LSR3. Защита линии является временной мерой, так как параллельно с началом использования обходного пути начальный узел основного пути начинает процедуру его восстановления с помощью протокола маршрутизации. После восстановления основного пути использование обходного пути прекращается. Временная защита линии не гарантирует TE-пути требуемой пропускной способности. Механизм защиты линии работает очень быстро, обычно время переключения не превосходит 50 мс, то есть сравнимо со временем переключения сетей SDH, которые всегда выступают в этой области в качестве эталона. Поэтому механизм защиты линии называют *быстрой перемаршрутизацией* (fats re-route).
- *Защита узла* очень похожа на защиту линии, отличаясь тем, что обходной путь прокладывается так, чтобы обойти отказавшее устройство LSR (в нашем примере это устройство LSR7). Все остальные характеристики аналогичны характеристикам защиты

линии; защита узла тоже относится к механизмам быстрой перемаршрутизации и тоже является временной мерой.

- *Защита пути* организуется так, что в дополнение к основному пути в сети прокладывается путь, связывающий те же конечные устройства, но проходящий, по возможности, через устройства LSR и линии связи, не встречающиеся в основном пути (на рисунке это резервный путь B3-LSP1). Данный механизм — самый универсальный, но он работает медленнее, чем механизмы защиты линии и узла.

Для быстрого обнаружения отказа основного пути или его части могут использоваться различные механизмы и протоколы: сообщения HELLO протокола RSVP, протокол LSP Ping или BFD.

Использование иерархии меток для быстрой защиты

Рассмотрим работу быстрых механизмов защиты на примере защиты линии, представленной на рис. 20.16. Пусть для защиты линии LSR2-LSR7 в сети проложен обходной путь B-LSP1. На основном пути LSP1 для продвижения кадров используется последовательность меток 15, 17 и 21. На первом участке обходного пути B-LSP1 используется метка 7, на втором — метка 8.

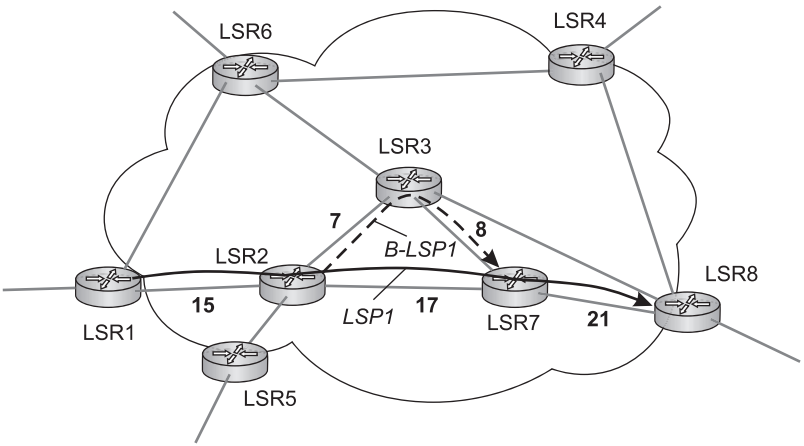


Рис. 20.16. Распределение меток для основного пути и обходного пути защиты линии

При отказе линии LSR2-LSR7 устройство LSR2 начинает направлять в обходной путь B-LSP1 кадры, поступающие по пути LSP1 (рис. 20.17). Но если при этом поменять метку 15 на метку 7, как того требует обычная логика коммутации меток, то кадр придет в устройство LSR7 с меткой 8 (ее установит устройство LSR3), которая не соответствует значению метки 17, используемой в устройстве LSR7 для передачи кадров по пути LSP1.

Чтобы устройство LSR7 работало при переходе на обходной путь точно так же, как и при нормальной работе основного пути, в технике быстрой защиты применяется иерархия

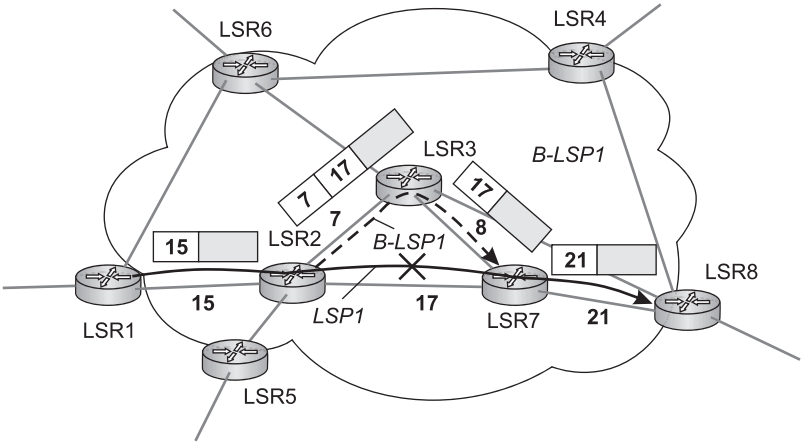


Рис. 20.17. Передача кадров по обходному пути

меток. Для этого устройство LSR2, которое реализует механизм защиты линии, заменяет метку 15 в пришедшем пакете меткой 17, как если бы линия LSR2-LSR7 оставалась работоспособной. Затем устройство LSR2 проталкивает метку первого уровня в стек, а на вершину стека помещает метку 7, которая нужна для продвижения кадра по обходному пути. Устройство LSR3 является предпоследним устройством обходного пути. Поэтому оно удаляет верхнюю метку 7 и выталкивает на вершину стека метку 17. В результате кадр поступает в коммутатор LSR7 с меткой 17, что и требуется для продвижения его далее по пути LSP1. Аналогичным образом работает механизм быстрой защиты узла, в нем также используется иерархия меток.

ГЛАВА 21 Виртуальные частные сети

Особенности VPN

Виртуальная частная сеть (Virtual Private Network, **VPN**) - это сервис, объединяющий сети одного предприятия или отдельных пользователей через публичную сеть, например интернет или сеть отдельного провайдера, и предоставляющий транспортные услуги, приближающиеся по качеству к услугам частной сети.

Определение «виртуальный» здесь имеет смысл «кажущийся» — пользователям этого сервиса кажется, что они изолированы от других пользователей публичной сети, что на их потоках данных мало сказывается существование потоков данных других пользователей, передаваемых через маршрутизаторы и каналы связи той же публичной сети. Можно сказать, что сервис виртуальных частных сетей *имитирует* сервис частной сети, но при этом он предоставляется публичной сетью с коммутацией пакетов.

Теперь давайте разберемся, что же такое «истинная» частная сеть, так как важно понимать, какие ее свойства имитируются ее виртуальной копией.

Истинная частная сеть

Термин «private» имеет два основных значения: частный (собственный) и конфиденциальный (закрытый). Если делать акцент на первом значении, то частная сеть — это такая сеть, в которой все оборудование (включая территориальные кабельные системы, коммутирующие устройства, средства управления и т. п.) является собственностью предприятия.

На рис. 21.1 приведен пример корпоративной сети, в которой связи между филиалами построены на основе собственного оборудования данного предприятия. Три территориальных канала связывают центральную локальную сеть предприятия с тремя локальными сетями удаленных офисов. Каждый территориальный канал образован отрезками кабеля, проложенного между устройствами регенерации, которые нужны для усиления сигналов и восстановления их формы после прохождения определенного расстояния по пассивному кабелю. Вся пропускная способность территориальных каналов находится в полном распоряжении предприятия.

Понятно, что сеть, построенная целиком на собственном оборудовании предприятия, соответствует и второму определению термина «private» — в собственной сети в принципе легче достигается конфиденциальность, поскольку все ресурсы сети используются только сотрудниками предприятия-владельца, а оборудование сети также обслуживается сотрудниками этого предприятия.

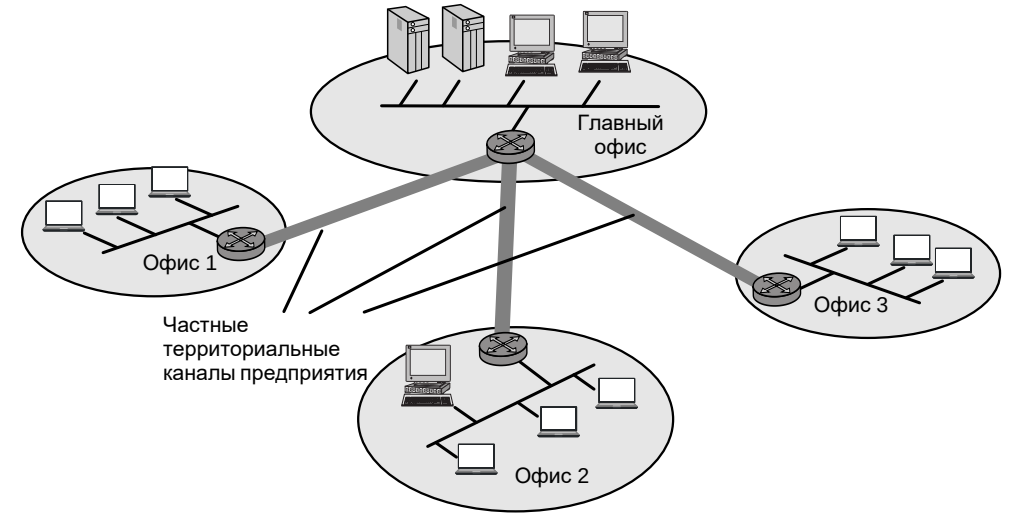


Рис. 21.1. Частная сеть предприятия с собственными территориальными каналами

Однако в мире существует немного корпоративных сетей, являющихся частными в таком «идеальном» смысле. Гораздо чаще предприятия используют для связи своих филиалов частные сети, построенные на арендуемых территориальных каналах. Эти каналы являются провайдером транспортных территориальных услуг в его первичной сети OTN/DWDM. На рис. 21.2 показан пример использования арендованных каналов для построения корпоративной сети уже рассмотренного нами предприятия с тремя филиалами. Каналы, связывающие центральную сеть предприятия с сетями филиалов, проходят через мультиплексоры и коммутаторы первичной сети.

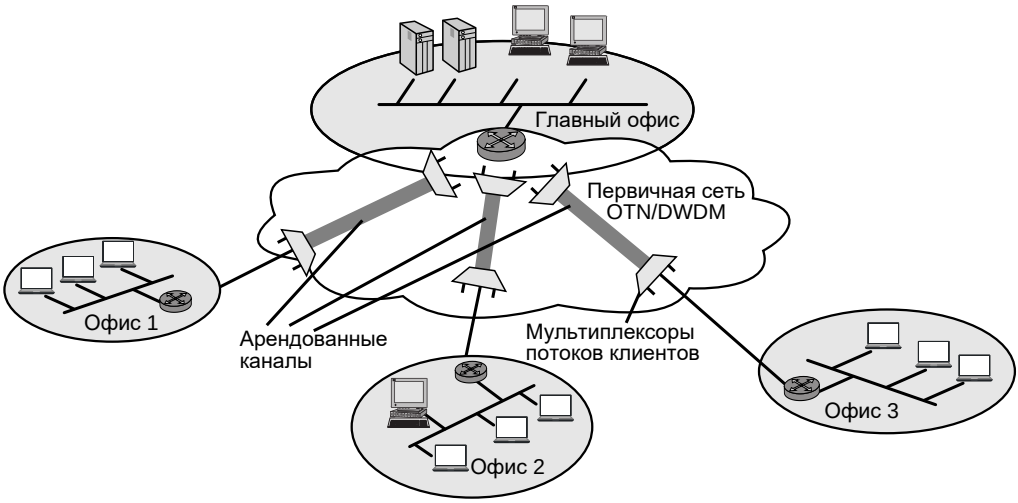


Рис. 21.2. Частные сети на основе арендованных каналов

Несмотря на то что территориальные каналы в этом случае уже не являются собственностью предприятия, корпоративные сети, построенные на арендованных каналах, также называются частными, и у них имеются по крайней мере две причины для того, чтобы именоваться таковыми. Во-первых, полоса пропускания арендованного канала полностью выделяется предприятию и поэтому является в некотором смысле его «частной собственностью». Из-за этого свойства такие каналы называются также *выделенными*. Наличие *гарантированной пропускной способности* дает возможность администратору корпоративной сети планировать работу приложений через глобальные каналы связи: распределять имеющуюся пропускную способность канала между приложениями, оценивать возможные задержки сообщений, ограничивать объем генерируемого территориального трафика, определять максимальное количество активных приложений и т. д.

Во-вторых, частный характер сетей, построенных на арендованных каналах, следует из того, что при передаче данных по таким каналам в значительной степени обеспечивается конфиденциальность. Корпоративные данные практически недоступны для других абонентов, не являющихся пользователями корпоративной сети или сотрудниками организации-провайдера каналов. Действительно, коммутацию каналов в первичных сетях может выполнить только оператор сети, пользователю сети такая операция недоступна. Это обусловливает большую степень защищенности данных, передаваемых по каналам первичных сетей. Например, здесь невозможна типичная для интернета атака — ответвление и анализ «чужого» трафика другим пользователем, что позволяет обеспечить приемлемую безопасность передаваемых данных без использования шифрования и взаимной аутентификации абонентов.

Несмотря на очевидные преимущества частной сети на арендованных каналах, у нее есть и существенные недостатки. Она по-прежнему слишком дорогая. Практически исключается возможность защищенного подключения к главному офису предприятия небольших филиалов и тем более индивидуальных пользователей, работающих из дома. Другим недостатком этого решения является отсутствие гибкости и масштабируемости: при дефиците каналов трудно создать сеть желаемой топологии.

Сервис виртуальных частных сетей использует публичный интернет или сеть с коммутацией пакетов одного провайдера для объединения отдельных подсетей и офисов предприятия (называемых также *сайтами*) в единую сеть. Гибкость, масштабируемость и экономическая эффективность такого решения очевидны. Очевидны и проблемы, которые возникают при использовании публичного интернета, — разнообразные атаки подвергают конфиденциальность и целостность передаваемых данных большому риску, кроме того, скорость передачи данных и их задержки не гарантированы.

Сервис виртуальных частных сетей решает эти проблемы за счет дополнительных технических приемов и технологий, при этом такой сервис стремится приблизиться к качеству сервиса частной сети в следующих областях:

- *Ограничение доступа к сети на уровне транспортных протоколов*: только сайты данной сети имеют техническую возможность посылать свои пакеты друг другу, взаимодействие с внешними пользователями ограниченно. Для технологии VPN обеспечить это свойство очень трудно, так как пакеты пользователей VPN проходят через те же коммуникационные устройства и каналы, что и пакеты внешних пользователей.
- *Независимая система адресации.* В частных сетях нет ограничений на выбор адресов — они могут быть любыми. Чтобы сохранить это свойство, сеть VPN должна допускать

адресацию узлов из всего диапазона IP-адресов, включая частные IP-адреса (рекомендованные только для автономного использования).

- *Предсказуемая производительность.* Собственные линии связи гарантируют заранее известную пропускную способность между узлами предприятия (для глобальных соединений) или коммуникационными устройствами (для локальных соединений). Обеспечение предсказуемой пропускной способности в публичной сети с коммутацией пакетов является целью сервиса VPN.
- *Максимально возможная безопасность.* Отсутствие связей с внешним миром ограждает частную сеть от атак извне и существенно снижает вероятность «прослушивания» трафика по пути следования пакетов. VPN ограничивает доступ внешних пользователей, а значит, исключает возможность атак извне, а для защиты от прослушивания можно применить шифрование.

Типы услуг VPN

Рассмотрим различные типы услуг VPN и поддерживающих их технологий. Услуги VPN отличаются набором свойств частной сети, которые они имитируют, и степенью приближения к качеству этих свойств.

1. *В зависимости от того, кто реализует услугу VPN* (провайдер или клиент), они подразделяются на два вида:

- В **поддерживаемой клиентом виртуальной частной сети** все тяготы по поддержке сети VPN ложатся на плечи пользователя. Провайдер предоставляет только «простые» традиционные услуги общедоступной сети по объединению узлов клиента, например доступ в интернет, а пользователи самостоятельно приобретают, устанавливают в своих сетях и конфигурируют средства VPN, а затем и поддерживают их работу.
- В случае **поддерживаемой провайдером виртуальной частной сети** провайдер услуг VPN на основе собственной сети воспроизводит частную сеть для каждого своего клиента, изолируя и защищая ее от остальных.

Эти два типа VPN иллюстрирует рис. 21.3. Виртуальная сеть предприятия *А* создана провайдером, а виртуальная сеть предприятия *Б* — силами специалистов самого предприятия *Б*. Средства VPN, поддерживающие образование сети VPN *А*, работают на маршрутизаторах сети провайдера, а средства, позволяющие организовать VPN *В*, установлены на пограничных маршрутизаторах сайтов предприятия *Б*.

Поддерживаемые провайдером сети VPN обычно обеспечивают более широкий спектр имитируемых свойств частной сети, чем поддерживаемые клиентом. Это объясняется тем, что провайдер имеет контроль над собственной сетью и может применить в ней соответствующую технологию и сконфигурировать свои устройства наиболее эффективным для оказания услуг VPN способом. Клиент такой возможности лишен. Он может использовать стандартный транспортный сервис провайдера и придать ему свойства VPN благодаря специальной конфигурации своих пограничных устройств.

Провайдер гарантирует конфиденциальность клиентского трафика от точки входа в публичную сеть до точки выхода, а также дает гарантии по качеству обслуживания клиентского трафика на этих же участках.

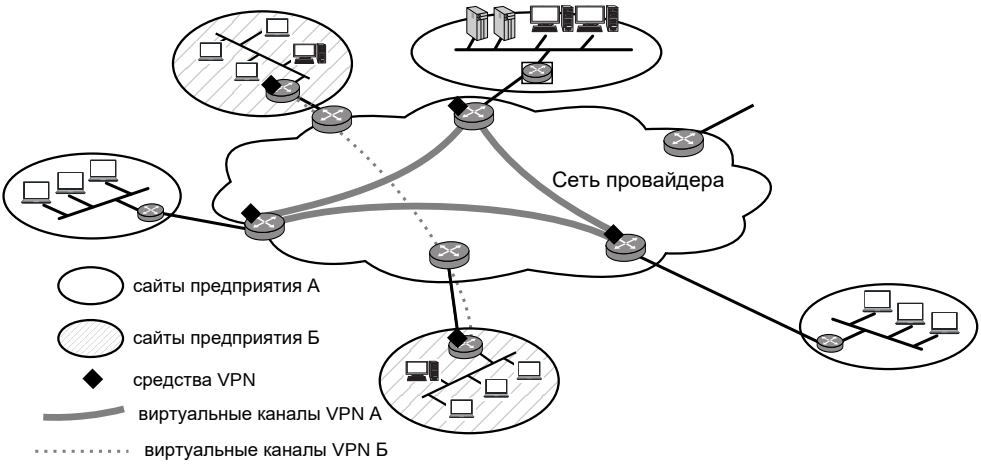


Рис. 21.3. Сети VPN двух предприятий

Во втором случае предприятие организует виртуальную частную сеть собственными силами, за счет применения специальных VPN-продуктов в своей сети. Правда, построить полноценную виртуальную частную сеть только своими силами, без участия провайдера, трудно. Все имеющиеся на рынке VPN-продукты обеспечивают решение только одной из двух необходимых для имитации частной сети задач, а именно выполняют защиту передаваемых данных на основе средств аутентификации и шифрации. Никаких же способов поддержания заданного качества транспортного обслуживания эти продукты не предоставляют. Это обстоятельство, однако, не говорит о незрелости VPN-продуктов — просто эти продукты делают то, что можно сделать, и не делают того, что в принципе сделать невозможно. Действительно, поддержать требуемое качество обслуживания для отдельных потоков данных в сети провайдера может только провайдер. Эта задача никак не может быть решена на клиентской стороне, и если провайдер не дает гарантий по обеспечению требуемого качества транспортного обслуживания, то нужно либо сменить провайдера, либо смириться с некоторой ущербностью своей VPN.

Поддержка гарантированного качества обслуживания — вещь достаточно сложная даже для провайдера. Обычно такое свойство достигается за счет применения средств инжиниринга трафика и применения механизмов QoS. Особенно сложно такие гарантии дать в том случае, когда виртуальная частная сеть организуется для сайтов, подключенных к сетям различных интернет-провайдеров, так как интернет в целом является сетью best effort, не гарантируя ни пропускную способность, ни параметры QoS. Поэтому услуги VPN, включающие не только защиту трафика от прослушивания и разрушения, но и гарантии транспортного обслуживания, могут быть предоставлены только в пределах той части интернета, которая поддерживается каким-либо одним провайдером, да и то таким, который предпринимает для этого в своей сети определенные меры.

2. *В зависимости от используемых технологий* виртуальные частные сети делятся на два класса:

- **сети VPN на основе разграничения трафика;**
- **сети VPN на основе шифрования**.

Виртуальная частная сеть на основе *разграничения трафика* использует создание каналов в публичной сети, которые работают на основе туннелирования пользовательского трафика. Такие туннели создают сеть, наложенную на публичную сеть: эта сеть и является виртуальной частной сетью. Говорят, что в публичной сети создается *оверлейная* (наложенная) структура. На рис. 21.3 оказаны три туннеля, созданные провайдером для построения VPN предприятия А. Как видно из рисунка, дополнительные средства VPN установлены на маршрутизаторах сети провайдера. Передача трафика пользователя VPN внутри туннеля изолирует его от трафика других пользователей этой сети и тем самым защищает от внешних атак и обеспечивает независимость адресных пространств сайтов VPN. Туннели этого типа можно назвать транспортными, потому что они транспортируют пользовательский трафик VPN отдельно от трафика других пользователей, в том числе по маршрутам инжиниринга трафика, что позволяет давать пользователю VPN некоторые гарантии QoS. Технологии создания туннелей VPN, обеспечивающих разграничение трафика в сети провайдера, могут быть различными. Одним из подходов является использование виртуальных соединений в сети IP/MPLS, эта техника рассматривается далее.

Виртуальная частная сеть на основе *шифрования* основана на использовании так называемых *защищенных каналов*. Такой канал не является транспортным туннелем. Его трафик передается через публичную сеть стандартным способом, без его инжиниринга, поэтому защищенный туннель может быть проложен через публичный интернет через произвольное число сетей провайдеров. Назначение защищенного канала состоит в защите пользовательских данных от прослушивания и разрушения, так как эти данные внутри канала передаются в зашифрованном виде. Даже если трафик этого канала и будет перехвачен внутри публичного интернета, то пользовательские данные злоумышленник прочесть не сможет, если они, конечно, зашифрованы достаточно надежно.

Защищенный канал может быть образован силами клиента интернета и от провайдеров обоих окончаний канала требуется только предоставление стандартного доступа в интернет. В этом состоит основное преимущество VPN на основе шифрования от VPN на основе разграничения трафика: первые работают в пределах всего интернета, в то время как вторые — в пределах сети одного провайдера, поддерживающего услугу VPN. На рис. 21.3 показаны два сайта предприятия *Б*, образовавшие сеть VPN за счет добавления к своим пограничным маршрутизаторам модулей создания защищенного канала.

Технологии защищенного канала могут работать на основе протоколов IPSec или TLS. Оба эти протокола рассматриваются в главе 31.

Сети VPN на основе шифрования могут быть организованы и силами провайдеров, но последний вариант распространен мало. Поэтому гарантий пропускной способности и QoS такие сети VPN, как правило, не предоставляют.

Для повышения уровня защищенности виртуальных частных сетей можно применять технологии VPN на основе разграничения трафика *совместно* с технологиями VPN на основе шифрования трафика — получив от провайдера услугу VPN на основе разграничения трафика, клиент самостоятельно может дополнить ее собственным защищенным каналом «из конца в конец», то есть между пограничными маршрутизаторами своих сайтов.

3. *В зависимости от того, адресная информация какого уровня принимается во внимание* при объединении сетей клиентов, различаются:

- **VPN второго уровня**: учитывается адресная информация второго (канального уровня) сетей клиентов, то есть MAC-адреса и идентификаторы VLAN;
- **VPN третьего уровня**: учитываются IP-адреса сетей клиентов.

VPN второго уровня объединяют локальные сети пользователя таким образом, как будто эти сети соединены коммутатором Ethernet. Только роль коммутатора здесь играет сеть провайдера, которая передает кадры Ethernet пользователя в транспортных туннелях, проложенных через IP-сеть. При этом IP-адресация сетей пользователя может быть произвольной, она целиком отдана на откуп администраторам этих сетей, и в этом состоит преимущество таких услуг VPN — пользователь такой услуги может менять структуру своей IP-сети, не координируя свои действия с провайдером, соединяющим эти сети в корпоративную сеть.

Услуги VPN второго уровня с интерфейсом Ethernet могут оказываться с помощью различных технологий:

- Технологий Ethernet операторского класса PB/PBB/PBB-TE.
- Технологий VPWS (топология «точка-точка») и VPLS (произвольная топология соединения сайтов) на основе туннелей IP/MPLS.
- Технологии BGP EVPN (Ethernet VPN), которая может работать как в сети IP/MPLS, так и в сети IP. В последнем случае для образования туннелей применяются такие технологии туннелирования, как VXLAN и GRE.

VPN третьего уровня объединяют IP-подсети пользователя в единую корпоративную IP-сеть. Такая услуга VPN обеспечивает независимость IP-адресов сайтов пользователя от IP-адресов глобального интернета за счет туннелирования пользовательского трафика. Поэтому часто сайты используют частные IP-адреса, что создает изоляцию сайтов VPN от публичного интернета.

Услуги VPN второго и третьего уровня относятся к услугам VPN на основе разграничения трафика. Основной технологией этого типа VPN является MPLS VPN.

Стандартизация услуг VPN второго уровня

Услуга VPN второго уровня с интерфейсом Ethernet является очень востребованной корпоративными клиентами. При этом она может оказываться на основе различных технологий, работающих в сети провайдера, отличаться поддерживаемыми топологиями и качеством обслуживания. Для того чтобы можно было сравнивать различные услуги этого важного класса VPN, некоммерческая организация Metro Ethernet Forum (**MEF**) разработала спецификацию услуги.

В спецификации MEF вводится три типа услуг виртуальных частных сетей Ethernet, которые отличаются *топологией* связей между сайтами пользователей. Для того чтобы формализовать топологию связей, вводится понятие **виртуального соединения Ethernet** (Ethernet Virtual Circuit, **EVC**). Каждое соединение EVC связывает сайты пользователей в отдельную виртуальную частную сеть, объединяя сетевые интерфейсы пользователей (UNI). Соответственно имеются три типа соединений EVC (рис. 21.4):

- «точка-точка» (двухточечная топология);
- «каждый с каждым» (полносвязная топология);
- «дерево» (древовидная топология).

Для каждого типа соединения определяется свой тип услуг:

- **E-LINE (услуга линии Ethernet)**. Эта услуга базируется на двухточечном EVC-соединении и заключается в объединении двух пользовательских сайтов.

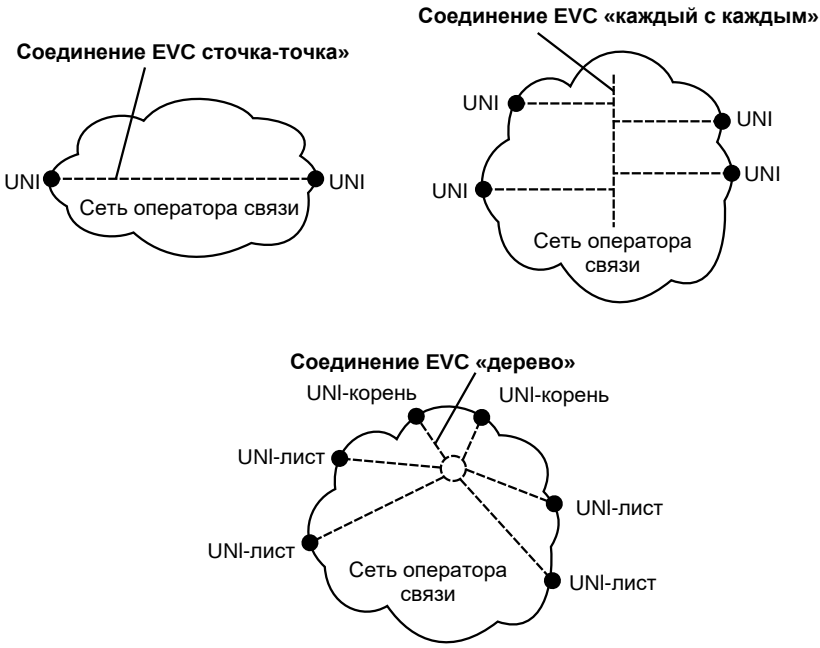


Рис. 21.4. Три типа услуг Ethernet

- **E-LAN (услуга локальной сети Ethernet)**. Эта услуга базируется на EVC-соединении «каждый с каждым» и позволяет связать неограниченное число пользовательских сайтов так, что каждый сайт может взаимодействовать с каждым. При этом соблюдается логика работы локальной сети — кадры Ethernet с неизученными и широковещательными MAC-адресами передаются всем сайтам, а кадры с изученными уникальными MAC-адресами — только тому сайту, в котором находится конечный узел с данным адресом.
- **E-TREE (услуга дерева Ethernet)**. Спецификация этой услуги появилась позже других; она базируется на EVC-соединении «дерево». Пользовательские сайты в этом случае делятся на корневые и листовые. Последние могут взаимодействовать только с корневыми, но не между собой. Корневые сайты могут взаимодействовать с листовыми сайтами и друг с другом. Такая услуга подходит для предприятия, у которого есть центральный офис, который становится корнем дерева, и удаленные офисы, играющие роль листьев.

В спецификации MEF вводятся два варианта каждого типа услуги. В первом варианте пользовательский сайт определяется как сеть, подключенная к отдельному физическому интерфейсу UNI. Значения идентификаторов VLAN в пользовательских кадрах (то есть значения C-VID в терминологии PB) в расчет не принимаются. В названии этого варианта услуги к названию типа добавляется термин «частный» (private). Например, для услуги типа E-LINE этот вариант называют **частной линией Ethernet** (Ethernet Private Line, **EPL**), а для услуги E-LAN — **частной локальной сетью Ethernet** (Ethernet Private LAN, **EPLAN**).

В другом варианте услуги к одному и тому же физическому интерфейсу UNI могут быть подключены различные пользовательские сайты. В этом случае они различаются по значению идентификатора VLAN (C-VID). Другими словами, провайдер внутри своей сети

сохраняет деление локальной сети на VLAN, сделанное пользователем. В варианте услуги с учетом VLAN добавляется название «виртуальная частная». Например, для услуги типа E-LINE это будет **виртуальная частная линия Ethernet** (Ethernet Virtual Private Line, **EVPL**), а для услуги E-LAN — **виртуальная частная локальная сеть Ethernet** (Ethernet Virtual Private LAN, **EVPLAN**).

В своих определениях MEF использует термины «частная услуга» и «виртуальная частная услуга» не совсем традиционным образом, так как оба типа услуги являются виртуальными частными в том смысле, что они предоставляются через логическое соединение в сети с коммутацией пакетов, а не через физический канал в сети с коммутацией каналов. Помимо указанных определений услуг спецификации MEF стандартизуют некоторые важные параметры услуг — например, услуга может характеризоваться гарантированным уровнем пропускной способности соединения, а также гарантированными параметрами QoS.

Виртуальные частные сети на базе MPLS

Предоставление сервисов **виртуальных частных сетей** (Virtual Private Network, **VPN**) является одной из основных областей применения технологии MPLS. Такие ее свойства, как логическое разделение потоков трафика разных пользователей на основе техники виртуальных каналов (в форме продвижения по меткам) и тесная интеграция с протоколами слоя управления стека TCP/IP, обеспечивают MPLS прочное место в спектре технологий VPN. Технология MPLS VPN предоставляет услуги VPN на основе сети MPLS провайдера.

Существуют два варианта технологи MPLS VPN:

- второго уровня (MPLS L2VPN), когда виртуальная частная сеть объединяет сети пользователя на канальном уровне, оперируя с MAC-адресами узлов этих сетей;
- третьего уровня (MPLS L3VPN), когда сети пользователя объединяются на уровне IP-адресов и IP-подсетей.

Оба варианта технологии MPLS VPN обеспечивают независимость адресного пространства сетей пользователя от адресов сети провайдера, которая объединяет сети пользователя.

Далее мы рассмотрим технологию MPLS VPN второго уровня, а описание технологии MPLS VPN третьего уровня вы можете найти на сайте книги.

Технология MPLS VPN второго уровня

Для использования MPLS как внутренней технологии провайдера при предоставлении услуг Ethernet VPN маршрутизаторы MPLS должны быть сконфигурированы специальным образом, а пограничные маршрутизаторы должны, кроме того, предоставлять пользователям интерфейсы Ethernet.

Псевдоканалы

Стандарты IETF описывают два типа услуг Ethernet VPN, которые строятся с помощью технологии MPLS: **VPWS** (Virtual Private Wire Service) и **VPLS** (Virtual Private LAN

Service). Различие между этими услугами в том, что VPWS эмулирует соединение Ethernet с двухточечной топологией, то есть канал Ethernet (соответствует услуге E-LINE в терминологии MEF), а VPLS эмулирует поведение локальной сети, то есть обеспечивает соединения с полносвязной топологией в стиле обычной локальной сети Ethernet (соответствует услуге E-LAN в терминологии MEF) .

Основным строительным элементом этих услуг являются так называемые **псевдоканалы**[1] (pseudowire), которые соединяют пограничные маршрутизаторы провайдера. На рис. 21.5 показано три таких псевдоканала, соединяющих между собой пограничные маршрутизаторы PE1–PE4 (PE — от Provider Edge).

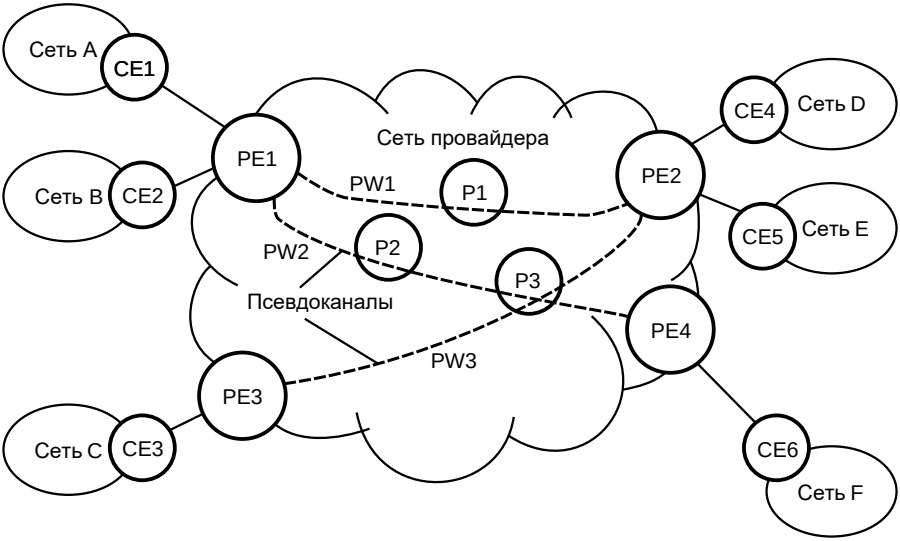


Рис. 21.5. Псевдоканалы в сети провайдера

Псевдоканалы представляют собой пути LSP второго уровня иерархии (называемого также внутренним уровнем), проложенные внутри LSP первого (внешнего) уровня. Обычно в качестве LSP первого уровня иерархии используются TE-туннели MPLS, так как они обладают полезными дополнительными свойствами, которых нет у путей, проложенных с помощью протокола LDP, например, более сбалансированной нагрузкой, что обеспечивает лучшее качество обслуживания. На рисунке пути LSP первого уровня не показаны, чтобы заострить внимание читателя на псевдоканалах.

Псевдоканалы — это логические транспортные соединения. Физически они могут проходить через промежуточные магистральные маршрутизаторы, однако для них они прозрачны, то есть в нашем примере маршрутизаторы P1, P2 и P3 просто не замечают их существования в сети. Однако псевдоканал — это не просто логическое соединение LSP второго уровня иерархии, поскольку, согласно определению, данному в RFC 3985, у псевдоканала есть более специфическое назначение.

---

[1] Встречаются и другие русские переводы термина pseudowire, например «эмулятор канала», «эмулятор кабеля», «псевдопровод».

Псевдоканал — это механизм, который эмулирует существенные свойства какого-либо телекоммуникационного сервиса через сеть с коммутацией пакетов.

Одним из вариантов применения псевдоканалов при эмуляции услуг Ethernet является передача псевдоканалом трафика одного пользовательского соединения, при этом псевдоканал эмулирует кабельное соединение между сетями пользователей. В примере на рисунке псевдоканал PW2 служит для организации соединения между сетями A и F через сеть провайдера. При этом кадры Ethernet, отправляемые сетью A в сеть F, инкапсулируются пограничным маршрутизатором PE1 в данные псевдоканала и доставляются им пограничному маршрутизатору PE4, который извлекает эти кадры и отправляет их в сеть F в первоначальном виде.

Из определения, данного в RFC 3985, видно, что назначение псевдоканала шире эмуляции Ethernet — это может быть и эмуляция сервисов выделенных каналов технологий первичных сетей PDH, SDH или OTN. Однако в любом случае эмуляция такой услуги выполняется через *пакетную сеть*. Тип пакетной сети также не уточняется, так что это может быть и классическая сеть IP (без MPLS), и сеть IP/MPLS. Главное в этом обобщенном определении — то, что псевдоканал скрывает от пользователей эмулируемого сервиса детали пакетной сети провайдера, соединяя пользовательские пограничные устройства (CE на рис. 21.5) таким образом, как если бы они соединялись с помощью выделенного канала или кабеля. Для некоторых наиболее важных сочетаний эмулируемого сервиса и типа пакетной сети комитет IETF разработал отдельные спецификации псевдоканалов. Далее мы рассмотрим тип псевдоканала, который нужен для предоставления услуг Ethernet операторского класса, а именно *псевдоканал эмуляции Ethernet* через сети IP/MPLS.

Технически создать LSP второго уровня достаточно просто — для этого в маршрутизаторах, соединенных LSP первого уровня, нужно задать значение метки второго уровня, которое будет использоваться, чтобы различать LSP второго уровня внутри LSP первого уровня (рис. 21.6).

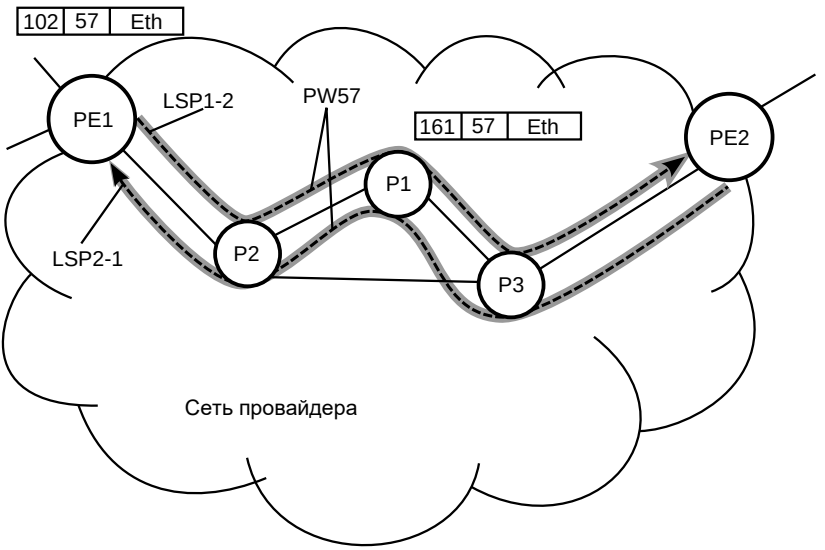


Рис. 21.6. Создание псевдоканала внутри туннелей MPLS

На нем изображены два пограничных маршрутизатора, PE1 и PE2, соединенные псевдоканалом PE57. Однако рисунок оказался немного сложнее, чем можно было предположить, — вместо одного пути LSP первого уровня мы видим два таких пути. Это связано с тем, что двухточечные псевдоканалы, которые служат для эмуляции Ethernet, по определению IETF всегда являются двунаправленными, а MPLS LSP — это однонаправленный путь. Поэтому для создания двунаправленного псевдоканала требуется два однонаправленных пути второго уровня, вложенных в два однонаправленных пути первого уровня, что и показано на рисунке.

Рассматриваемый в нашем примере псевдоканал в направлении от PE1 к PE2 идентифицируется MPLS-меткой 57, а туннель, который использует этот канал, — MPLS-меткой 102. Поэтому при отправке кадра Ethernet, предназначенного для PE2, маршрутизатор PE1 помещает исходный кадр Ethernet в кадр MPLS и адресует этот кадр двумя метками: внешней меткой 102 и внутренней меткой 57. Внешняя метка применяется затем магистральными маршрутизаторами P1, P2 и P3 для того, чтобы доставить кадр пограничному маршрутизатору PE2, при этом в процессе передачи кадра происходит обычная коммутация по меткам (на рисунке показано, что после прохождения P1 внешняя метка получила значение 161). Внутренняя метка 57 требуется только пограничному маршрутизатору PE2, который знает, что эта метка соответствует псевдоканалу PW57, необходимому для связи с некоторой пользовательской сетью.

Как мы видим из рассмотренного примера, псевдоканалы работают только внутри сети провайдера, так что для эмуляции сервиса «из конца в конец» нужны еще какие-то элементы и механизмы — и мы скоро их рассмотрим, но сначала давайте обсудим преимущества применения псевдоканалов поверх MPLS. Возникает естественный вопрос: нельзя ли обойтись LSP первого уровня для передачи трафика Ethernet через сеть провайдера, не используя концепцию псевдоканала? В принципе, без псевдоканалов обойтись можно, но тогда для каждого нового пользовательского соединения пришлось бы создавать новый туннель (то есть LSP первого уровня), а это не очень масштабируемое решение, так как конфигурирование такого пути обязательно подразумевает конфигурирование всех магистральных маршрутизаторов сети. Поэтому одно из существенных преимуществ псевдоканалов состоит в том, что в сети провайдера нужно сконфигурировать только сравнительно небольшое число туннелей между пограничными маршрутизаторами, а затем использовать каждый из них для прокладки необходимого числа псевдоканалов. Создание нового псевдоканала также требует конфигурирования, но только пары пограничных маршрутизаторов, являющихся конечными точками псевдоканала, что подразумевает гораздо меньший объем работы.

Другим преимуществом псевдоканалов является их универсальность, то есть возможность их применения не только в сетях MPLS, но и в сетях других типов, таких как чистые IP-сети с туннелированием (например, по протоколу L2TP или GRE), и не только при эмуляции Ethernet, но и при эмуляции других сервисов (например, каналов OTN). Естественно, что при переходе к другой реализации псевдоканалов конкретные команды конфигурирования меняются, но концепция остается, и это помогает администраторам сети освоить новую технологию.

Услуги VPWS

Услуги **виртуальных частных каналов** (Virtual Private Wire Service, **VPWS**) исполняют роль «глобального кабеля», соединяя прозрачным образом две локальные пользовательские сети Ethernet через сеть оператора связи. Мы рассмотрим организацию такой услуги

с помощью псевдоканалов MPLS на примере (рис. 21.7). При этом мы опишем дополнительные элементы механизма эмуляции услуги Ethernet, которые были опущены при описании псевдоканалов.

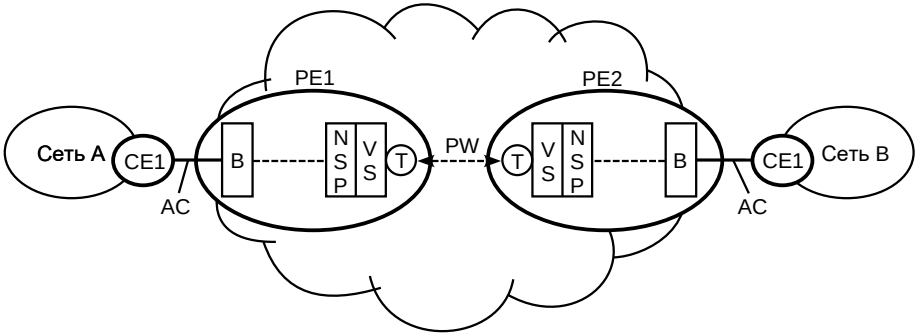


Рис. 21.7. Организация виртуального частного канала Ethernet

Чаще всего пользовательские сети соединяются с пограничным маршрутизатором провайдера через выделенный интерфейс, который для глобальных услуг Ethernet должен быть стандартным интерфейсом Ethernet. В этом случае услуга VPWS заключается в прозрачном соединении этих интерфейсов, когда сеть провайдера передает все кадры, поступающие на такой интерфейс от сети пользователя. Иногда этот режим VPWS называют коммутацией портов пользователя (в терминологии MEF это услуга EPL). Возможен и другой вариант услуги VPWS, когда сеть провайдера соединяет виртуальные пользовательские сети, то есть по двухточечному соединению передаются не все кадры, поступающие через интерфейс пользователя, а только кадры, принадлежащие определенной сети VLAN. Этот режим работы VPWS можно назвать коммутацией виртуальных локальных сетей, или VLAN-коммутацией (в терминологии MEF это услуга EVPL).

Чтобы обобщить понятие интерфейса с пользователем, было введено понятие **канала присоединения** (Attachment Circuit, **AC**). AC поставляет входной поток пользовательских данных для сети провайдера, то есть ту нагрузку, которую нужно коммутировать. Употребляя этот термин, можно сказать, что услуга VPWS всегда соединяет два пользовательских канала присоединения.

На рисунке показаны также внутренние функциональные элементы пограничных маршрутизаторов PE1 и PE2, эмулирующих услуги VPWS вместе с псевдоканалом PW. Модуль *B* (от англ. *B*ridge — мост) работает по стандартному алгоритму моста. Его роль в схеме эмуляции — выделение кадров Ethernet из общих потоков, поступающих на порты маршрутизатора, для передачи в псевдоканал. Тем самым модуль моста формирует логический интерфейс виртуального коммутатора. Например, если это режим коммутации портов, то модуль моста конфигурируется так, чтобы все кадры, пришедшие на соответствующий порт от пользователя, направлялись для дальнейшей обработки в псевдоканал. Если же это VLAN-коммутация, то модуль моста выбирает для передачи псевдоканалу только кадры, помеченные определенным значением тега VLAN.

Выбранные модулем моста кадры поступают в псевдоканал не непосредственно, а через два промежуточных модуля — NSP и VS. Модуль NSP (Native Service Processing) обеспечивает предварительную обработку кадров Ethernet. Чаще всего такая обработка связана

с изменением или добавлением тега VLAN, что может потребоваться, например, если объединяемые пользовательские сети применяют различные значения VLAN. Модуль VS (Virtual Switch — виртуальный коммутатор) коммутирует один из каналов присоединения с одним из псевдоканалов. Для услуги VPWS этот модуль работает вхолостую, выполняя постоянную коммутацию единственного канала присоединения с единственным псевдоканалом. Но для услуги VPLS, рассматриваемой в следующем разделе, виртуальный коммутатор играет важную роль, поэтому в обобщенной схеме эмуляции услуг Ethernet, представленной на рис. 21.7, он присутствует.

После обработки пришедшего кадра модулями NCP и VS он передается псевдоканалу. Конечные точки *T* псевдоканала PW57 выполняют две операции:

- инкапсуляцию и декапсуляцию пользовательских кадров в кадры MPLS;
- мультиплексирование и демультиплексирование псевдоканалов в туннеле MPLS.

При инкапсуляции у исходного кадра отбрасываются поля преамбулы и контрольной суммы, после чего он помещается в кадр MPLS с двумя полями меток: внешней (метка туннеля) и внутренней (метка псевдоканала), как это показано на рис. 21.8.

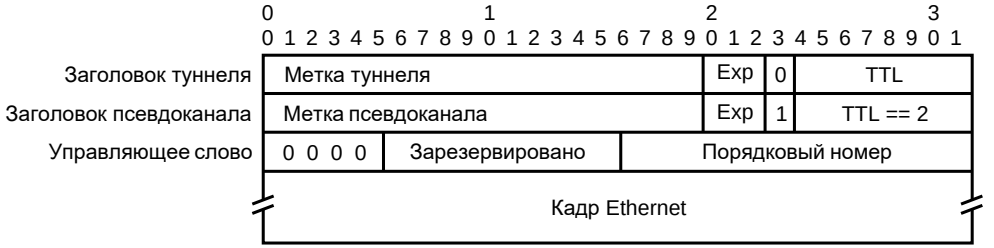


Рис. 21.8. Формат инкапсуляции Ethernet поверх MPLS

В то время как первые два слова в заголовке, представленном на рисунке, являются стандартными заголовками MPLS, третье слово, называемое *управляющим* (control word), впервые появилось в стандарте VPWS. Оно необязательно и предназначено для упорядочивания кадров, передаваемых по псевдоканалу, — для этого каждому кадру маршрутизатором-отправителем присваивается порядковый номер, помещаемый в управляющее слово. Потребность в управляющем слове возникает, когда внутри сети провайдера происходит распараллеливание трафика туннеля и кадры могут выходить из туннеля не в том порядке, в котором были направлены в него.

Конфигурирование псевдоканалов, то есть согласование внутренних меток, используемых для идентификации и мультиплексирования псевдоканалов внутри туннеля, может быть автоматизировано. Для этого сегодня применяют протокол LDP или BGP. Обратите внимание, что речь идет о прокладке псевдоканала, а не самого туннеля (эти два процесса независимы), так что туннель может быть проложен, например, с помощью протокола SR-MPLS, а псевдоканалы в нем — с помощью протокола LDP.

В завершение описания услуг VPWS хочется напомнить, что такое важное свойство услуги, как гарантированная пропускная способность, обеспечивается с помощью техники инжиниринга трафика, опирающейся в данном случае на соответствующие свойства туннелей MPLS. Аналогично обстоит дело с параметрами качества обслуживания (QoS) для виртуальных соединений VPWS — они могут быть обеспечены с помощью стандартных

механизмов QoS: приоритетное обслуживание, профилирование трафика, контроль доступа и резервирование ресурсов. И в этом случае MPLS является хорошим базисом, так как детерминированность маршрутов туннелей MPLS делает контроль доступа намного более определенной процедурой, чем в случае IP-сетей с их распределенным (и вносящим неопределенность) механизмом выбора маршрутов.

Услуги VPLS

Услуги **виртуальной частной локальной сети** (Virtual Private LAN Service, **VPLS**) позволяют объединить в виртуальную частную сеть несколько сетей клиентов. При этом в сети провайдера может существовать несколько виртуальных частных сетей клиентов, например, VPLS1, VPLS2, VPLS3 и т. д.

Как и в случае VPWS, сервис VPLS организован на базе псевдоканалов. Отличие состоит в том, что для каждого экземпляра VPLS используется *отдельный набор псевдоканалов*. Каждый такой набор имеет полносвязную топологию, то есть все пограничные маршрутизаторы PE, участвующие в работе какого-то экземпляра VPLS, связаны друг с другом.

На рис. 21.9 показан пример сети провайдера, эмулирующей две виртальные частные сети VPLS. Пользовательские сети C1, C5 и C8 относятся к «серому» сервису VPLS, а сети C2, C3, C4, C6 и C7 — к «белому». Соответственно, набор псевдоканалов PW-B1, PW-B2 и PW-B3 объединяет пограничные маршрутизаторы, к которым подключены сети «серого» сервиса VPLS, а набор псевдоканалов PW-W1, PW-W2 и PW-W3 — маршрутизаторы, к которым подключены сети «белого» сервиса VPLS (в нашем примере это одни и те же пограничные маршрутизаторы PE1, PE2 и PE3, но если бы сети C4 не существовало, то псевдоканалы PW-W2 и PW-W3 были бы не нужны).

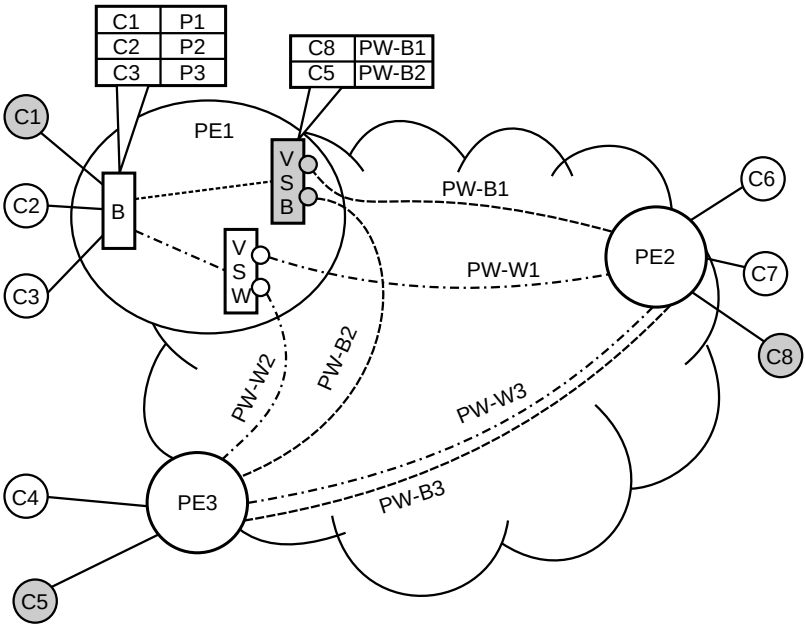


Рис. 21.9. Организация услуги VPLS

Внутренняя организация пограничного маршрутизатора при оказании услуги VPLS показана на примере маршрутизатора PE1. Мы видим, что для поддержки каждого экземпляра сервиса VPLS пограничному маршрутизатору требуется *отдельный виртуальный коммутатор*, в данном случае — модули VSB и VSW (модули NSP не показаны, чтобы не загромождать рисунок, но они в PE1 входят, по одному на каждый экземпляр VPLS). Как и в случае VPWS, модуль *B* выполняет стандартные функции моста и при этом формирует логический интерфейс с каждым из виртуальных коммутаторов. Этот интерфейс может также формироваться на основе коммутации либо пользовательских портов, когда весь трафик от определенного порта (или нескольких портов) передается на логический интерфейс, либо сетей VLAN, когда выбираются кадры одной (или нескольких) пользовательских сетей VLAN от одного или нескольких портов.

Однако если в случае VPWS виртуальный коммутатор выполнял простую работу по передаче кадров от логического интерфейса, то для VPLS этот модуль функционирует по алгоритму стандартного коммутатора (моста). Для этого виртуальный коммутатор изучает MAC-адреса и строит свою таблицу продвижения, как и обычный коммутатор. На рисунке показан упрощенный вид таблицы продвижения PE1, состоящей из двух записей: одна запись связывает адрес M8 сети C8 с псевдоканалом PW-B1, другая — адрес M5 сети C5 с псевдоканалом PW-B2. Посредством таблицы виртуальный коммутатор не затапливает сеть, получая кадры с адресами M5 или M8, а направляет их в псевдоканал, ведущий к пограничному коммутатору, к которому подключена сеть с узлом назначения. Кадры с широковещательным адресом или адресом, отсутствующим в таблице продвижения, поступают на все его псевдоканалы, в данном случае — на PW-B1 и PW-W1.

> Единственной особенностью виртуального коммутатора является то, что он не изучает адреса отправления кадров, приходящих с логического интерфейса.

Эта операция не нужна, потому что для интерфейсов, представленных псевдоканалами, виртуальный коммутатор работает по *правилу расщепления горизонта* (split horizon) — он никогда не передает на псевдоканал кадры, полученные от какого бы то ни было псевдоканала. Тем самым предотвращается образование петель между виртуальными коммутаторами, а доставку кадров по назначению гарантирует полносвязная топология. То есть любой кадр, полученный виртуальным коммутатором по псевдоканалу, всегда передается на логический интерфейс пользователя, соответствующий тому сервису VPLS, к которому относится псевдоканал. Модуль моста *B* изучает только адреса, приходящие с пользовательских интерфейсов, служащие ему для выбора нужного интерфейса в том случае, когда несколько пользовательских сетей относятся к одному сервису VPLS.

Существует возможность создания виртуальных частных сетей 2-го уровня с интерфейсом Ethernet на основе протокола BGP. Такие сети получили название BGP EVPN (Ethernet VPN). Эта технология не использует технику псевдоканалов и позволяет создавать сеть EVPN не только на базе сети MPLS, но и на базе сети IP (без MPLS), используя различные технологии туннелирования трафика, например GRE или VXLAN.

Виртуальные частные сети 3-го уровня на базе MPLS позволяют соединять несколько IP-сетей клиента на основе IP-адресов через сеть MPLS, их описание вы можете найти на сайте книги.

(S) *Технология MPLS VPN третьего уровня*

Вопросы к части V

1. Какие из перечисленных услуг относятся к информационным (выберите вариант ответа):
   а) выделенные каналы OTN;
   б) интернет-телефония;
   в) веб-хостинг.
2. К какому типу сети оператора связи должен быть подключен центр данных, управляющий роботами в реальном масштабе времени (выберите вариант ответа):
   а) к магистрали;
   б) к сети доступа.
3. Верно ли утверждение: «Услуги виртуальных частных сетей являются информационными услугами»?
4. Периферийные вычисления выполняются на (выберите вариант ответа):
   а) компьютере пользователя;
   б) сервере провайдера, подключенном к его сети доступа;
   в) сервере провайдера, подключенном к его магистральной сети.
5. Пиринговое соглашение между интернет-провайдерами оговаривает (выберите вариант ответа):
   а) взаимную блокировку нелегального контента;
   б) взаимный обмен трафиком.
6. Какой узел является инициатором обмена данными:
   а) клиент;
   б) сервер;
   в) маршрутизатор.
7. Верно ли следующее утверждение: «Протокол HTML переносит страницы, размеченные по правилам языка HTTP»?
8. Браузер находит информацию по адресам специального формата, например, такому: http://www.bbc.co.uk/mobile/web/versions.shtml. Поясните, какая часть данной записи представляет собой:
   а) путь к объекту;
   б) DNS-имя сервера;
   в) UR-имя;
   г) тип протокола доступа.

9. Что вы можете сказать о HTTP-сообщении вида HTTP/1.1 200 OK? Варианты ответов:
   а) это HTTP-запрос;
   б) это HTTP-ответ;
   в) 200 — это код состояния;
   г) 200 — это объем переданной информации;
   д) OK означает, что информация зашифрована открытым ключом;
   е) OK означает, что «все в порядке!».
10. Укажите, какие из следующих утверждений правильны:
   а) веб-сервер находится постоянно в активном состоянии, прослушивая TCP-порт 80;
   б) веб-сервер переходит в активное состояние, получив запрос от клиента на TCP-порт 80;
   в) после отправки всех объектов клиенту сервер разрывает с ним TCP-соединение;
   г) веб-сервер не хранит информацию о том, какие страницы пользователь уже посетил;
   д) получив запрос от клиента, веб-сервер отображает все объекты на экране.
11. За счет чего в версии HTTP/2 достигается ускорение процесса построения веб-браузером веб-страницы по сравнению с HTTP 1.0 и 1.1:
   а) работы поверх более быстрого протокола UDP;
   б) конвейеризации запросов и ответов;
   в) использования параллельных TCP-соединений.
12. Верно ли следующее утверждение: «Основная проблема версий HTTP 1.0, 1.1 и 2 заключается в блокировке начала очереди»?
13. Почему версия протокола HTTP 3 не использует протокол TCP:
   а) потому что для ускорения работы она перешла на более быстрый протокол UDP;
   б) функции протокола TCP по надежной доставке данных реализованы самим протоколом HTTP 3.
14. В чем состоят недостатки организации почтовой службы, при которой почтовый клиент и сервер установлены на компьютере у каждого абонента:
   а) компьютер пользователя может стать слишком загружен работой почтового сервера;
   б) компьютер пользователя должен быть все время активен, если же он выключен, то письмо этому пользователю нельзя отправить;
   в) имя компьютера пользователя необходимо регистрировать в системе DNS, чтобы почтовые клиенты других пользователей могли его найти, — плохая масштабируемость.
15. Агент системы управления сетью — это:
   а) программа, устанавливаемая на управляемое устройство;
   б) аппаратный блок, встраиваемый производителем в управляемое устройство;
   в) аппаратный модуль, способный измерять характеристики проходящего через управляемое устройство трафика;

г) программа, способная по командам выполнять конфигурирование управляемого устройства;

д) программно-аппаратный модуль, управляющий вычислительным процессом в устройстве.

16. Что такое MIB в системе управления сетью? Варианты ответов:

а) модель управляемого объекта;

б) база данных управляющей информации;

в) протокол взаимодействия агента и менеджера системы управления сетью;

г) набор характеристик объекта, необходимых для его контроля.

17. Какую роль играет агент в двухзвенной схеме управления устройством? Варианты ответов:

а) роль клиента;

б) роль сервера;

в) роль посредника.

18. Какой протокол может быть использован для взаимодействия менеджера и агента системы управления сетью? Варианты ответов:

а) SNMP;

б) SMTP;

в) MIP;

г) FTP;

д) Telnet.

19. Назовите преимущества техники виртуальных каналов:

а) быстрое продвижение кадров за счет небольших размеров таблиц продвижения (сокращается время просмотра таблицы);

б) гибкость маршрута;

в) защищенность от злоумышленников;

г) небольшие размеры заголовка кадра.

20. Перечислите функциональные модули IP-маршрутизатора, которые используются в LSR:

а) модуль маршрутизации;

б) модуль продвижения пакетов на основе IP-адресов;

в) модуль ARP-протокола.

21. Предположим, что LSR использует формат кадров Ethernet. Означает ли это, что LSR продвигает кадры на основе таблицы продвижения, полученной в соответствии со стандартом IEEE 802.1D?

22. Протокол LDP является частью технологии:

а) MPLS IGP;

б) MPLS TE.

23. Можно ли в сети, поддерживающей MPLS, передавать часть трафика с помощью обычного IP-продвижения?
24. Технология MPLS является гибридом технологий:
    а) IP и IPX;
    б) IP и OSPF;
    в) IP и технологии виртуальных каналов.
25. Класс эквивалентности продвижения — это:
    а) набор путей LSP с равными метриками;
    б) набор путей к одному и тому же выходному LER;
    в) группа IP-пакетов, имеющих одни и те же требования к условиям транспортировки.
26. Какой из вариантов управления распределением меток протоколом LDP называется упорядоченным:
    а) метка назначается по запросу от вышележащего устройства LSR;
    б) метка не назначается устройством LSR до тех пор, пока оно не получит метку от нижележащего устройства;
    в) метка назначается администратором сети.
27. Какой механизм отказоустойчивости MPLS является более универсальным (выберите вариант ответа):
    а) защита линии;
    б) защита пути.
28. Стек меток MPLS используется для:
    а) задания иерархии путей LSP;
    б) защиты путей LSP;
    в) маршрутизации от источника.
29. Верно ли утверждение: «Инжиниринг трафика в технике SR-MPLS является более масштабируемым, чем с помощью протокола RSVP»?
30. Может ли технология SR-MPLS работать в режиме обычного продвижения пакетов на основе их IP-адресов?
31. Может ли сегмент маршрута технологии SR-MPLS задать операцию компрессии заголовка пакета?
32. В чем проявляется глобальность идентификаторов узлов Node-SID:
    а) они глобальны в масштабе интернета, как и публичные IP-адреса;
    б) они глобальны в пределах отдельного TE-пути;
    в) они глобальны в пределах сети MPLS.
33. Всегда ли можно задать TE-путь только с помощью идентификаторов узлов Node-SID?

34. Список SR-Policy может быть сформирован:
    а) пограничным маршрутизатором LER;
    б) внешним узлом;
    в) промежуточным маршрутизатором LSR.
35. Почему вариант контроллера SR-MPLS на основе протоколов BGP LS и BGP SR является более масштабируемым, чем на основе протокола PCEP:
    а) потому что протокол BGP хорошо зарекомендовал себя в сетях провайдеров интернета;
    б) потому что этот вариант не требует установления сессий между контроллером и всеми узлами сети SR-MPLS;
    в) потому что контроллер в этом случае может быть установлен в любой подсети интернета, даже в другой автономной системе.

Часть VI

Беспроводная передача данных

- Глава 22. Особенности беспроводных сетей
- Глава 23. Технологии физического уровня беспроводных сетей
- Глава 24. Сотовые и спутниковые сети
- Глава 25. Технологии WI-Fi и Bluetoth
- Глава 26. Беспроводные технологии с низким энергопотреблением
- Глава 27. Сети дронов

В главе 22 рассматриваются проблемы, с которыми мы раньше не сталкивались в этой книге и которые характерны для беспроводных сетей: мобильность как норма, нестабильность линий связи, быстрые изменения топологии сети, учет энергозатрат при разработке протоколов всех уровней и некоторые другие. В этой главе также дается детальная классификация типов беспроводных сетей.

Все технологии беспроводных сетей базируются на общем фундаменте, образуемом совокупностью технологий физического уровня, которые рассматриваются в главе 23. Здесь рассматриваются диапазоны и характер распространения электромагнитных волн, виды искажений и методы борьбы с ними, различные типы антенн, включая антенны MIMO. В дополнение к описанным в главе 7 методам кодирования, одинаково применимым к проводным и беспроводным сетям, в данной главе рассматриваются технологии кодирования расширенного спектра FHSS, DSSS, CDMA и OFDM/OFDMA, которые были разработаны специально для беспроводной передачи.

В главе 24 рассматриваются технологии мобильных сетей: четвертого поколения LTE, в которой мобильная сеть стала полностью IP-сетью, и пятого поколения 5G, которая предложила новые скорости доступа к интернету и поддержку самых разнообразных типов услуг — для автоматизации технологических процессов, управления роботами и беспилотными автомобилями, а также многочисленных сервисов для пользователей интернета вещей. Здесь также изучаются спутниковые сети разного типа, включая системы Iridium и Starlink.

Главы 25 посвящена беспроводным локальным сетям Wi-Fi (IEEE 802.11), которые в секторе фиксированного беспроводного доступа к интернету заняли такую же доминирующую позицию, что и сети Ethemet в локальных сетях, В этой главе также рассматривается технология Bluetooth, позволяющая многочисленным гаджетам, находящимся на небольших расстояниях, взаимодействовать друг с другом.

В главе 26 изучаются технологии низкого энергопотребления, ориентированные на применения в различных областях интернета вещей. Здесь изучаются как технологии малого радиуса действия, такие как ZigBee и Thread, так и большого радиуса действия, такие как LoRaWAN и Wi-SUN FAN.

Последняя глава этого раздела посвящена сравнительно новому типу сетей — сетям дронов (FANET). Рассматриваются различные типы архитектур этих сетей и специфические радиотехнологии для построения связей в такой сети. Делается акцент на взаимосвязи навигационной и коммуникационной систем сети дронов.

ГЛАВА 22 Особенности беспроводных сетей

Ключевые вехи в развитии беспроводных сетей

7 мая 1895 года в Санкт-Петербурге на заседании физического отделения Русского физико-химического общества выступил 35-летний физик Александр Степанович Попов. Он продемонстрировал работу своего прибора, который обнаруживал излучение радиосигналов, посылаемых передатчиком, на расстоянии до 60 м и в принципе мог быть использован для передачи информации. Завершил А. С. Попов свою презентацию следующими словами: «В заключение могу выразить надежду, что мой прибор при дальнейшем усовершенствовании может быть применен к передаче сигналов на расстояние при помощи быстрых электрических колебаний, как только будет найден источник таких колебаний, обладающий достаточной энергией».

Впереди лежал долгий путь от первой радиопередачи до первой беспроводной компьютерной сети.



Рис. 22.1. А. С. Попов представляет первое радио на заседании Русского физико-химического общества

В 1971 году в Гавайском университете была представлена **первая беспроводная компьютерная сеть** с коммутацией пакетов **ALOHA**. Она послужила основой для создания протоколов Ethernet и Wi-Fi CSMA/CA (подробнее об этом см. главы 10 и 25). На ранних этапах развития компьютерных сетей большая часть применений беспроводной связи была связана с ее фиксированным вариантом, в форме радиорелейных или спутниковых линий связи, связывающей центры присутствия операторов сети.

К концу 70-х была разработана концепция сотовой связи и внедрена **первая коммерческая аналоговая сотовая сеть** первого поколения 1G (NTT Япония, 1979). Аналоговые сотовые сети были предназначены только для голосовой связи, использовали частотную модуляцию (ЧМ) и не имели шифрования.

Возможность передавать информацию без проводов, привязывающих (в буквальном смысле этого слова) абонентов к определенной точке пространства, всегда была очень привлекательной. Первый мобильный телефон был изобретен еще в 1910 году Ларсом Магнусом Эрикссоном (Lars Magnus Ericsson). Этот телефон предназначался для автомобиля и был беспроводным только во время движения. Однако в движении им нельзя было пользоваться — для разговора нужно было остановиться, выйти из автомобиля и с помощью длинных жердей присоединить телефон к придорожным телефонным проводам (рис. 22.2). Понятно, что определенные неудобства и ограниченная мобильность воспрепятствовали коммерческому успеху этого вида телефонии.

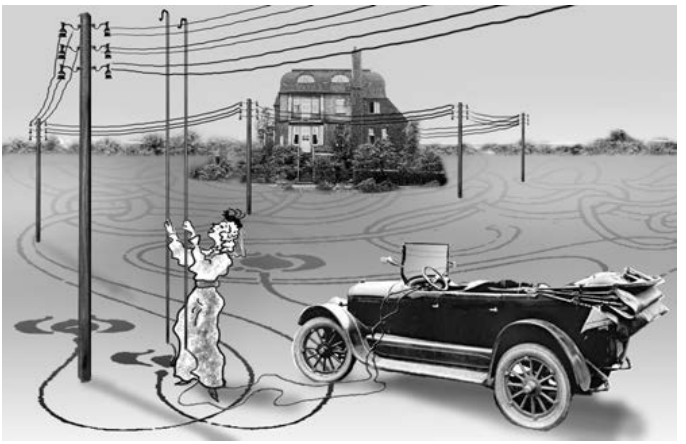


Рис. 22.2. Первый мобильный телефон

Прошло много лет, прежде чем технологии радиодоступа достигли определенной степени зрелости и в конце 70-х обеспечили производство сравнительно компактных и недорогих радиотелефонов. С этого времени начался бум мобильной телефонии, продолжающийся до настоящего времени.

1991 год ознаменовался запуском первого стандарта **цифровой сотовой связи** — стандарта GSM (2G), который поддерживал услуги SMS и мобильной передачи данных.

Развитие технологии сотовых телефонных сетей привело к тому, что эти сети стали очень широко использоваться для мобильного доступа к интернету и, начиная с поколения 4G, стали скорее мобильными компьютерными сетями, чем мобильными телефонными сетя-

ми, так как основной услугой в них стал доступ к интернету, а телефонные услуги стали предоставляться также с использованием протокола IP и интернета.

С появлением стандарта IEEE 802.11 в 1997 году стало возможным строить мобильные сети Wi-Fi, по функциональности близкие к Ethernet и обеспечивающие как фиксированный, так и мобильный доступ пользователей к интернету. Применение антенн с множественными потоками данных (MIMO) и мультиплексирования OFDMA (2019 г., стандарт Wi-Fi 6/802.11ax) позволило резко увеличить плотность пользователей и скорость передачи данных в густонаселенных областях — стадионах, аэропортах, концертных и конференцзалах и т. п.

Последнее поколение мобильных сетей 5G достигло скоростей 1–10 Гбит/с и также стало оказывать услуги с очень низкой задержкой сигнала до 1 миллисекунды. Такая функциональность поддерживает технологический прорыв в областях, требующих практически мгновенной реакции: управление беспилотными автомобилями, системы виртуальной реальности, телемедицина, робототехника.

В конце этой главы приведена классификация беспроводных сетей, включающая сети разных масштабов (от самоорганизующихся небольших сетей Bluetooth до спутниковых сетей Starlink), разной степени мобильности (статические сенсорные сети LoRaWAN — летающие сети FANET), разных технологий (Wi-Fi, ZigBee, 4G LTE, 5G) и др. Все это многообразие сетей может служить прекрасной иллюстрацией успехов эволюции беспроводных технологий.

Далее в этой главе мы рассмотрим особенности беспроводных сетей. Беспроводные компьютерные сети — это частный, хотя и очень важный случай компьютерных сетей, поэтому нужно понимать, что большая часть того, что написано в предыдущих главах, справедлива и в отношении беспроводных сетей. Беспроводная сеть также состоит из узлов и линий связи, узлы имеют адреса и имена, коммуникационные протоколы беспроводных сетей образуют многоуровневый стек, соответствующий модели OSI, сети с различными технологиями физического и канального уровней объединяет протокол IP, в ячеистых топологиях работают протоколы маршрутизации.

Однако есть и коренное отличие: в проводной сети узлы соединяются проводными (кабельными) линиями связи, а в беспроводных — воздушной средой или космическим вакуумом. Именно отсюда следуют все остальные особенности беспроводных сетей: повышенная мобильность узлов, изменчивая топология, разделяемая среда как доминирующий способ доступа, нестабильность и пониженная пропускная способность линий связи, а также сложность поддержания безопасности.

Нестабильность линий связи

Нестабильность характеристик линий связи является базовым свойством беспроводных сетей.

В проводных сетях качество линий связи известно заранее, многие характеристики, такие, например, как помехоустойчивость, затухание сигнала в определенном диапазоне частот, полоса пропускания, электрический импеданс и др., гарантированы стандартом на кабель.

В беспроводных же сетях качество передающей среды может быстро меняться из-за разного рода помех. Одной из помех является сама Земля: нет прямой видимости на больших

расстояниях. Кроме того, на меньших расстояниях могут мешать дома, деревья и горы, пролетающие объекты. Атмосферные явления — дождь, туман, облака — также создают помехи. Источником помех являются и созданные человеком источники электромагнитных волн, например мачты мобильных сетей. Всем знакомо переменное качество телефонного разговора по мобильной сотовой сети. Такие помехи могут превратить двустороннюю связь в одностороннюю или вообще привести к отказу линии связи. А главное, во многих случаях предсказать изменения качества беспроводной передающей среды невозможно. Очевидно, что мобильность узлов еще больше увеличивает нестабильность и снижает производительность линий связи в беспроводной сети.

В то же время нельзя утверждать, что проводная связь по своей природе более скоростная, чем беспроводная связь. Представим себе связь двух лазеров в идеальной беспроводной среде — в вакууме. При отсутствии помех скорость обмена данными будет выше, чем у многих типов проводных линий связи. Так, NASA в 2022 году смогла достичь скорости передачи данных с низкоорбитального спутника (высота орбиты около 500 км) на земную станцию со скоростью 200 Гбит/с с помощью лазерной коммуникационной системы (мировой рекорд для оптической спутниковой связи на момент написания данной книги).

На практике же беспроводные каналы связи уступают проводным по пропускной способности в десятки и даже сотни раз. Например, технология 5G декларирует максимальную пропускную способность 20 Гбит/с, Wi-Fi 6E — до 9,6 Гбит/с. В то же время самая скоростная технология Ethernet имеет намного более высокую скорость — 800 Гбит/с, а соединения OTN — 400 Гбит/с и выше.

Беспроводные линии связи будут продолжать иметь значительно более низкую пропускную способность по сравнению с проводными аналогами. Реальная пропускная способность беспроводной коммуникации — после учета влияния множественного доступа, затухания, шума, интерференции и т. д. — значительно ниже максимальной скорости передачи радиосигнала. Одним из последствий относительно низких или умеренных пропускных способностей является то, что перегрузки обычно являются нормой, а не исключением, то есть совокупный запрос приложений может часто превышать пропускную способность сети.

Нестабильность линий связи приводит к невозможности простого переноса коммуникационных технологий из мира проводных сетей в мир беспроводных сетей. Менее надежные воздушные каналы вынуждают разработчиков использовать, например, более короткие кадры и увеличенные тайм-ауты по сравнению со стандартными протоколами проводных сетей. В беспроводных сетях используются адаптивные методы кодирования и модуляции, приспособленные к постоянному изменению характеристик среды, устойчивые к помехам, — например, технологии расширения спектра.

Разделяемая среда

Понятие разделяемой среды является одним из первых, с которых начинается изучение компьютерных сетей. Технология разделяемой среды как средство построения компьютерной сети была впервые использована при создании сети ALOHA — прародительницы технологии Ethernet, после чего на базе разделяемой среды было создано многочисленное поколение проводных сетей Ethernet на коаксиальном кабеле и повторителях. Однако

относительно скоро эволюция проводных компьютерных сетей привела к исчезновению разделяемой среды как средства связи нескольких узлов сети.

Иное дело беспроводные сети, в которых разделяемая среда по-прежнему играет основную роль и все устройства используют одну и ту же беспроводную среду (воздух) для передачи данных.

Разделяемая среда — это любая физическая среда, которая может передавать через себя независимые сигналы от нескольких источников. Последовательность сигналов, которыми обмениваются два узла-источника, называется потоком данных. Примеры разделяемых сред:

- одна медная пара;
- оптическое волокно;
- блок радиочастотного спектра;
- беспроводной канал в космическом пространстве.

Разделяемая радиосреда — это область распространения радиосигналов, такая, что все узлы сети, которые находятся в этой области, могут слышать друг друга и благодаря этому совместно использовать ее для обмена данными.

Хотя весь радиоэфир потенциально является единой разделяемой средой, он может состоять из нескольких изолированных разделяемых подсред, так что группа узлов, работающих в одной подсреде, может слышать только друг друга и не может обмениваться данными с другой группой узлов, работающих в другой подсреде. Каждый поток данных воспринимает подсреду как свой собственный канал.

Разделение общей разделяемой среды на изолированные подсреды происходит с помощью техники **мультиплексирования**. Мы уже рассматривали некоторые виды мультиплексирования, в частности, частотное FDM, временно́е TDM (синхронное и асинхронное) и волновое WDM/DWDM-мультиплексирование. В этой части книги вы познакомитесь и с некоторыми другими типами мультиплексирования.

Поясним на примере связь между мультиплексированием, разделяемой средой и подсредами. Для того чтобы слышать друг друга в сети, использующей мультиплексирование FDM, узлы должны работать в одном и том же частотном канале и использовать одну и ту же технику модуляции. Например, узлы, работающие в частотном канале 2,4–2,42 ГГц с модуляцией 16-QAM, могут обмениваться данными между собой, и для них эта часть радиоэфира является разделяемой средой. В то же время узлы, работающие в диапазоне 5,7–575 ГГц и использующие ту же модуляцию 16-QAM, могут общаться между собой, но не с узлами первой группы.

В этом примере различные частотные диапазоны представляют собой различные разделяемые среды или, используя терминологию проводных сетей, — различные сегменты. Для связи узлов из разных сегментов в беспроводных сетях так же, как и в проводных сетях, нужно использовать коммуникационные устройства мосты/коммутаторы или маршрутизаторы.

Методы доступа к разделяемой среде в беспроводных сетях делятся на *централизованные*, на основе некоторого центрального элемента сети, выполняющего функции арбитра, и *децентрализованные*, адаптированные к возможному отсутствию связности между удаленными узлами сети (например, метод доступа Ethernet на коаксиальном кабеле CDMA/CD из-за этого не может применяться в беспроводной локальной сети, потребовалась его модификация).

Зона покрытия и топология

Зоной покрытия (coverage zone) узла беспроводной сети является некоторая область пространства вокруг этого узла, в которой распространяются его сигналы. Мощность излучаемого узлом сигнала уменьшается по мере удаления от этого узла. Если этой мощности достаточно для того, чтобы некоторый узел мог обмениваться информацией с узлом-источником сигнала, то говорят, что он находится в зоне покрытия или в зоне досягаемости этого узла. Область покрытия имеет границу, за пределами которой мощность и качество излучаемых сигналов не достаточны для того, чтобы приемники других узлов его принимали.

На рис. 22.3, *а* показана зона покрытия узла *A*, в которой находятся беспроводные узлы *B1, B2, B3*, то есть узел *A* излучает радиосигнал, достаточный для установления надежной связи с узлами *B1, B2, B3*. Узел *C* находится вне зоны покрытия узла *A*. Зоны покрытия узлов *B1, B2, B3* не позволяют им всем непосредственно связываться друг с другом, например, узел *B1* может связываться с *B2* или *B3* только через узел *A*, в то же время узлы *B2* и *B3* могут установить между собой прямую связь. Действительно, два узла могут установить связь, если каждый из них входит в зону покрытия другого. Узел *C*, хотя и не входит в зону покрытия узла *A*, может связываться с узлами сети опосредованно, через узел *B1*, а следовательно, узел *C* является частью сети (рис. 22.3, *б*).

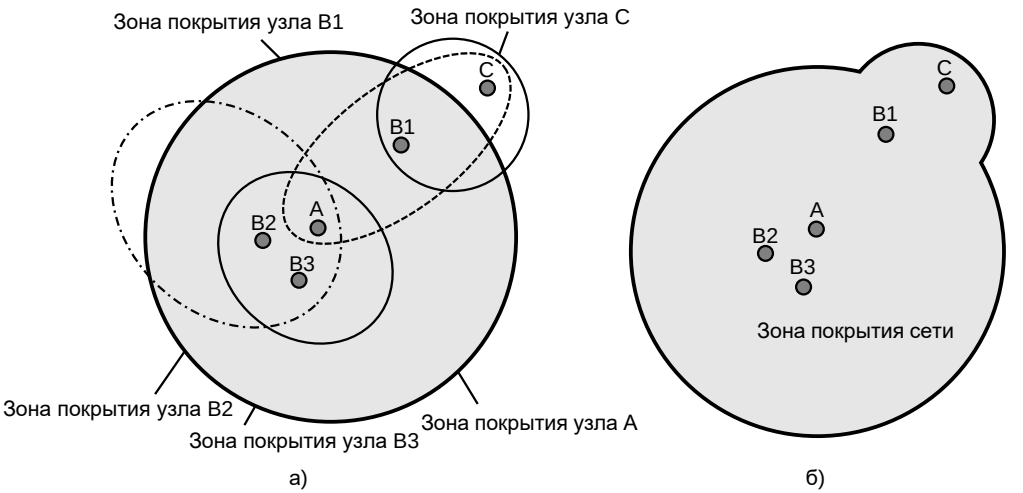


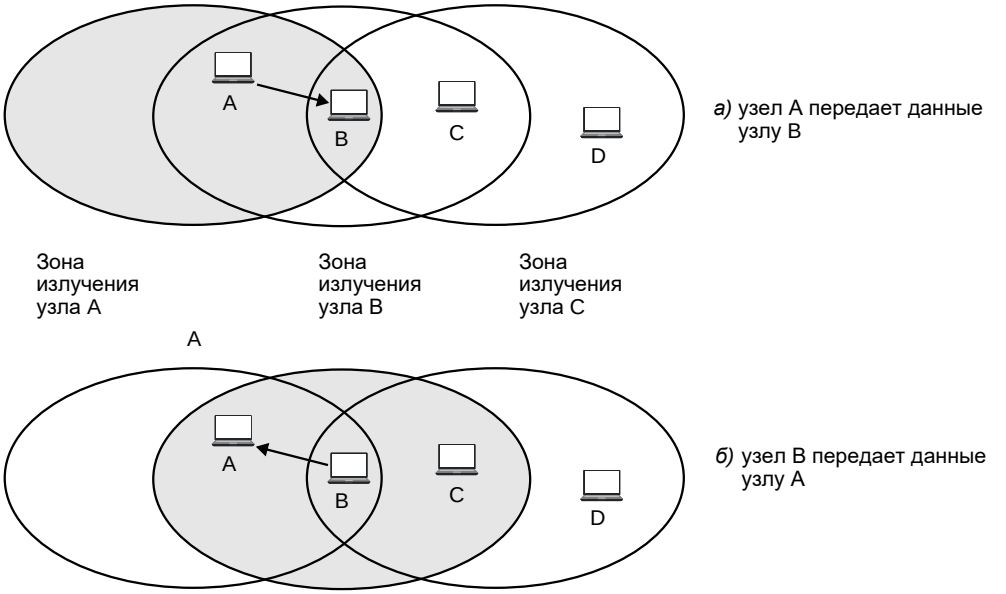
Рис. 22.3. а) зоны покрытия отдельных узлов сети; б) зона покрытия всей сети

Область пространства, в которой сигнал от узлов беспроводной сети достаточно силен для того, чтобы устройства, расположенные в этой области, могли устанавливать надежные соединения и получать доступ к этой сети, называется **зоной покрытия** беспроводной сети.

Размер и конфигурация зоны покрытия сети зависит от следующих факторов:

- взаимного пространственного расположения всех узлов сети;
- мощности передатчиков и чувствительности приемников узлов сети;
- типа и направленности антенн;
- характеристик среды передачи, включая наличие помех и препятствий.

Очевидно, что область покрытия узла может изменяться со временем, особенно в мобильной сети. Часто используемое изображение такой области в форме шестиугольника или круга является не чем иным, как абстракцией. Область покрытия даже не обязательно является связанной областью и может включать «мертвые» зоны, в которых связь отсутствует.



а) узел A передает данные узлу B

б) узел B передает данные узлу A

Рис. 22.4. Проблемы «скрытой» станции и «Засвеченной» станции

В примере на рис. 22.4, *а* показана такая фрагментированная локальная сеть. Неполносвязность беспроводной сети порождает проблемы доступа к разделяемой среде, известные как проблемы скрытой станции и засвеченной станции. **Проблема скрытой станции** возникает, если два узла находятся в зоне досягаемости друг друга (узлы *A* и *C* на рис. 22.4, *а*), но существует третий узел — *B*, который принимает сигналы как от *A*, так и от *C*. Предположим, что в радиосети используется *традиционный метод доступа, основанный на прослушивании несущей, например CSMA/CD*. В данном случае коллизии будут возникать значительно чаще, чем в проводных сетях. Пусть, например, узел *A* передает информацию узлу *B*. Узел *C* видит, что среда свободна, и начинает передавать свой кадр узлу *B*. В результате сигналы в районе узла *B* исказятся, то есть произойдет коллизия, вероятность возникновения которой в проводной сети была бы неизмеримо ниже. Другая ситуация показана на рис. 22.4, *б*. Узел *B* передает данные узлу *A*, в это время узел *C* решает связаться с узлом *D*, но, проверив среду, решает, что она занята, хотя на самом деле узел *D* доступен. Эта ситуация называется **проблемой засвеченной станции**.

В некоторых беспроводных сетях выделяется один-два узла, имеющих зону покрытия, охватывающую все узлы сети. Примером может являться точка доступа AP в сети Wi-Fi. В других сетях, состоящих из идентичных узлов, зона покрытия сети является результатом взаимного наложения зон покрытия отдельных узлов.

Аналогичной характеристикой беспроводных сетей является **дальность связи** (communication range) — максимальное расстояние, в пределах которого два узла могут надежно обмениваться данными. Дальность связи существенно меняется от технологии к технологии. Так, в технологии LoRa (о которой будет сказано далее в главе 26) идет речь о километрах, а в Bluetooth — о метрах.

Подобно проводным сетям, беспроводные сети также имеют топологию связей, однако в отличие от проводной сети, в которой связь между парой узлов может образовываться только путем фиксированного физического соединения их отрезком кабеля, в беспроводной сети топологические связи между двумя узлами образуются динамически за счет взаимного наложения их областей покрытия. В данном случае мы имеем дело с **физической топологией**. Физическая связь между двумя беспроводными узлами существует, если существует потенциальная возможность обмена информацией между этими узлами.

Однако такая возможность не всегда является востребованной и желаемой — например, физически связанные беспроводные узлы могут принадлежать сетям разных организаций, или же их взаимодействие не предусмотрено архитектурой сети. Если же наличие связи узлов соответствует логике работы системных и прикладных задач сети, то на основе физического соединения выполняется логическое соединение, обычно заключающееся в выполнении коммуникационной процедуры в соответствии с тем или иным протоколом. Совокупность логических связей беспроводной сети называется **логической топологией** сети.

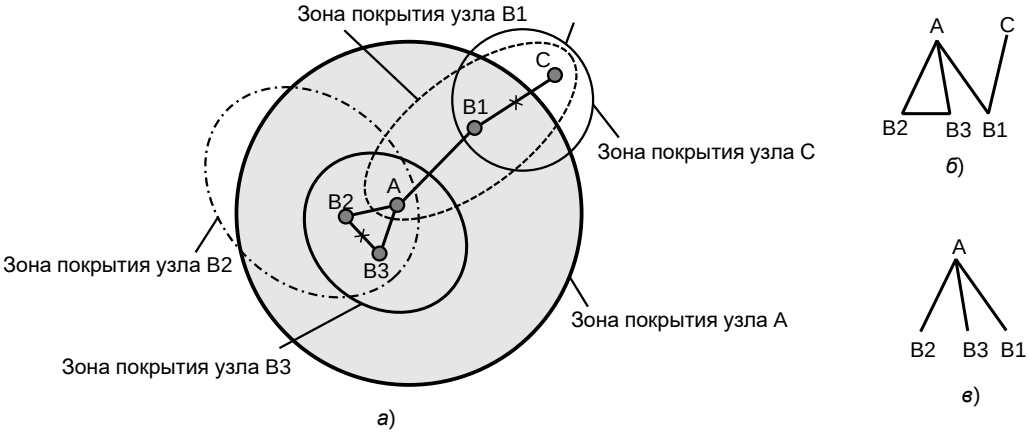


Рис. 22.5. а) физические связи узлов сети; б) физическая топология сети, в) логическая топология сети

На рис. 22.5 показана сеть Wi-Fi с одной точкой доступа (узел *A*) и несколькими клиентами (узлы *B1, B2, B3* и *C*), имеющими перекрывающиеся зоны досягаемости, что потенциально позволяет им взаимодействовать друг с другом. Однако в этой сети поддерживается логическая топология «звезда», при которой клиенты могут связываться друг с другом только через точку доступа *A* (такое решение имеет определенные преимущества, мы их рассмотрим далее). Поэтому топология логических связей этой сети (рис. 22.5, *в*) отличается от топологии ее физических связей (рис. 22.5, *б*): в нее не вошли связи между узлами *B2* и *B3*, а также *B1* и *C*. Логическая связь *B1-B2* не была установ-

лена из-за запрета прямого общения между конечными узлами в сети Wi-Fi, имеющей точку доступа, а связь *B1-C* потому, что узел *C* не принадлежит данной сети Wi-Fi как не входящий в зону покрытия точки доступа *A*.

В то же время существует и другой режим работы сети Wi-Fi, в котором используются физические связи между клиентами, что превращает логическую топологию «звезда» в логическую ячеистую топологию, при которой любой узел может взаимодействовать с любым другим узлом сети непосредственно, не нуждаясь в передаче данных через центральный элемент в виде точки доступа.

В беспроводных сетях мы встречаем те же виды топологии, что и в проводных сетях:

- *Точка-точка* — простейший случай, например, управление с земли одиночным дроном.
- *Полносвязная топология*, при которой каждый узел непосредственно связан с каждым другим.
- *Звезда* — топология, при которой одно устройство выступает в качестве центрального концентратора, а другие устройства подключаются к нему напрямую, например в сетях Wi-Fi.
- *Сетчатая, или ячеистая* (mesh), топология, в которой каждый узел связан с несколькими другими узлами, что создает структуру с избыточными соединениями для передачи данных. Может быть получена путем устранения некоторых связей из полносвязной структуры. Это повышает отказоустойчивость.
- *Частично ячеистая* топология — это комбинация ячеистой структуры и «звезды», где некоторые устройства связаны с большим количеством других устройств, в то время как некоторые — только с небольшим.
- *Иерархическая звезда (дерево)* широко используется в мобильных сотовых сетях.
- *Иерархическая ячеистая* топология может встречаться в сетях управления неоднородным роем дронов.

Если все беспроводные узлы находятся в пределах досягаемости друг друга, то соответствующая сеть имеет полносвязную топологию. Полносвязная топология обладает ценным качеством — любая пара узлов сети с такой топологией связана напрямую, то есть любой адрес назначения достигается из любого узла за один прыжок (hop), а значит, отпадает проблема маршрутизации. Такие сети называют **одношаговыми** (single-hop).

В сети любой другой топологии всегда найдется пара узлов, кратчайшее расстояние между которыми пролегает через один или более промежуточных узлов. В таких сетях, называемых **многошаговыми** (multi-hop), для передачи данных необходима маршрутизация.

В некоторых приложениях поддержание одношаговой топологии оказывается неэффективным, так как требует высоких затрат мощности на покрытие всех узлов сети, особенно если некоторые узлы размещаются далеко от основной группы. В условиях ограниченной пропускной способности беспроводных каналов предпочтительнее многошаговая топология связей.

Заметим, что в некоторых многошаговых сетях, в которых путь продвижения данных заранее известен, сеть также может работать без маршрутизации. Обратимся еще раз к рис. 22.5. Сеть имеет логическую топологию «звезда», а значит, связь между клиентами *B1*, *B2*, *B3* осуществляется за два шага, однако маршрутизация в данном случае не нужна, так как передача данных между ними всегда выполняется через один и тот же промежуточный узел — точку доступа *A*.

Типы узлов в беспроводных сетях

Подобно проводной сети, беспроводная сеть состоит из конечных узлов разного типа, которые соединены друг с другом коммуникационными связями, в данном случае эти связи беспроводные. Набор типа узлов зависит от конкретной архитектуры сети: является ли она сетью WI-Fi, телекоммуникационной сетью или сетью дронов.

Конечные узлы/рабочие станцин/хосты — это устройства, генерирующие данные, которые коммуникационная сеть должна передать либо другим конечным узлам этой же сети, либо конечным узлам другой, связанной с ней сети (возможно, проводной). На конечных узлах беспроводной сети так же, как и на конечных узлах проводной сети, работают разнообразные приложения, как клиентские, так и серверные. Конечные узлы могут быть как мобильными, так и фиксированными (стационарными). В качестве конечных узлов выступают различные типы компьютеров — ноутбуки, десктопы, лэптопы, смартфоны и т. п. Как правило, с конечными узлами ассоциируются конечные пользователи, которые используют ресурсы этих узлов для выполнения локальных задач, а также для передачи запросов к другим узлам сети и получения ответов на эти запросы из сети. Конечными узлами могут быть устройства, не предусматривающие пользователей-людей, такие как внешние устройства, например принтеры, разнообразные бытовые прибора и гаджеты, сенсорные датчики и исполнительные механизмы, дроны, беспилотные автомобили и др.

В тех случаях, когда конечные узлы не связаны друг с другом напрямую, для работы сети необходимо наличие коммуникационной сети. Коммуникационная сеть состоит из узлов, в задачу которых входит распознавание потоков данных (например, по адресам назначения), мультиплексирование/демультиплексирование потоков, а также определение маршрутов и продвижение данных в соответствии с найденными маршрутами. Эти задачи выполняют **маршрутизаторы**, работающие на сетевом уровне, соединяющие разные IP-сети. Помимо коммуникационных функций маршрутизаторы могут выполнять роль серверов DHCP, NAT. Функции коммуникационных узлов беспроводной сети могут выполняться теми же устройствами, которые используются в качестве конечных узлов. Например, функции маршрутизатора могут выполняться смартфоном, компьютером или дроном.

Беспроводная сеть состоит из одного или более сегментов. Все узлы, принадлежащие к одному сегменту, должны работать в одном и том же частотном радиодиапазоне и использовать тот же способ кодирования, то есть все узлы сегмента должны быть оснащены согласующимися сетевыми интерфейсами — приемниками и передатчиками радиосигналов с антеннами (их называют также **радио**).

В то же время беспроводная сеть может быть гетерогенной, то есть объединять в себе сегменты, в которых взаимодействие узлов осуществляется с использованием различающихся радиосигналов или даже разных сетевых технологий, в том числе проводных. Для связывания сегментов в беспроводных сетях используются устройства канального уровня — **мосты/коммутаторы**. Мост позволяет проводной сети подключаться к беспроводной сети или подключать несколько беспроводных устройств к одной проводной сети.

Пусть в беспроводной сети часть узлов работают в диапазоне 2,4 ГГц, а другая часть — диапазоне 5 ГГц, то есть они используют разные разделяемые среды, и, следовательно, образуют два разных беспроводных сегмента. Для передачи кадров между ними нужен классический прозрачный мост/коммутатор, работающий по тому же алгоритму, что и мост/коммутатор проводных сетей.

В тех же случаях, когда разделяемый сегмент один и в нем имеется центральный узел (базовая станция LTE/5G или точка доступа Wi-Fi), через который передаются кадры всех узлов сети, такой центральный узел работает как нестандартный, специфический мост. Его специфика состоит в том, что он возвращает кадры, полученные из сегмента в тот же самый сегмент, чего стандартный мост делать не должен (как было объяснено в главе 11, он их просто отбрасывает).

Важную роль в беспроводных сетях играют **повторители/ретрансляторы** — узлы, которые расширяют область покрытия путем усиления и улучшения качества полученного сигнала, например extender сети Wi-Fi. Заметим, что работа беспроводного повторителя, как и работа моста, отличается от работы его аналога в проводной сети. Вместо повторения поступившего на вход сигнала бит за битом, как это делает проводной повторитель, беспроводной повторитель буферизует весь кадр, декодирует его, выполняя в некоторых случаях проверку контрольной суммы, а затем передает весь кадр центральному узлу сети (например, точке доступа Wi-Fi) через тот же самый сегмент разделяемой беспроводной сети. То есть повторитель беспроводной сети работает в **режиме хранения-продвижения** (store-and-forward), и в этом отношении он аналогичен мосту/коммутатору проводной сети. Однако «мостом» такое устройство можно назвать весьма условно, так как в беспроводной сети оно не делит разделяемую среду на сегменты и не отбрасывает кадр, полученный от одного конечного узла беспроводной сети и адресованный другому конченому узлу того же сегмента, как это должен был бы сделать мост проводной сети. Нет, после буферизации повторитель передает его в тот же сегмент, из которого кадр получен.

Для беспроводных сетей характерно наличие так называемых гибридных узлов, совмещающих в одном устройстве несколько функций, функциональные аналоги которых отсутствуют в проводных сетях. Таким узлом, например, является узел-**лидер кластера** (cluster head) в сети дронов, функцией которого является координация группы узлов, агрегирование и передача группового трафика. Лидер кластера часто обладает более мощными ресурсами, чем остальные узлы группы.

Другим примером является **базовая станция** (base station) сотовой сети, антенна которой взаимодействует с мобильными телефонами посредством радиоволн, подключая их к мобильной сети.

Гибридным многофункциональным узлом также является **точка доступа** (AP, access point) в сетях Wi-Fi, которая функционирует как промежуточное звено между клиентами беспроводной сети и проводной сетью. В зависимости от конфигурации сети Wi-Fi точка доступа может работать в режиме моста между беспроводной и проводной частью сети, или как комбинация устройств мост/повторитель, мост/маршрутизатор. Точка доступа может также выполнять функции арбитра, управляя распределением пропускной способности радиосреды между конечными узлами в том случае, когда методы мультиплексирования позволяют это сделать.

Инфраструктурные сети и сети без инфраструктуры

Инфраструктурной сетью (infrastructure network) называется сеть, в которой можно выделить две части — набор конечных узлов и сетевую инфраструктуру.

Сетевая инфраструктура — это комплекс программ, аппаратных средств, кабелей и устройств, формирующих сеть. Инфраструктура по отношению к конечным узлам играет вспомогательную роль. Именно благодаря инфраструктуре конечные узлы получают возможность взаимодействовать, согласованно работать, совместно использовать ресурсы сети.

Приставка «инфра» на латинском языке обозначает «находящийся ниже» (например, инфракрасный, инфразвук). То есть инфраструктура сети представляет собой заранее предустановленную поддерживающую среду для обеспечения совместной работы конечных узлов. Инфраструктура разделяется всеми конечными узлами сети, предоставляя всем узлам сети свои услуги.

К основным компонентам сетевой инфраструктуры относятся:

- различные выделенные серверы, такие как почтовый сервер, веб-сервер, серверы DHCP, файловый сервер, сервер бизнес-приложения, сервер аутентификации пользователей и др.;
- принтеры и другие внешние устройства, подключенные непосредственно к сети (не через компьютер);
- коммуникационной оборудование — маршрутизаторы, коммутаторы, мультиплексоры, шлюзы.

Могут ли существовать сети без инфраструктуры? Да, могут. Такие сети называются **безынфраструктурными сетями** (infrastructure less network). Если не принимать во внимание кабели, соединяющие компьютеры, то примером сети без инфраструктуры является простейшая локальная проводная сеть Ethernet на общей шине. Другим примером является одношаговая беспроводная сеть, все узлы которой находятся в пределах досягаемости друг друга, в такой сети маршрутизация не требуется, и сеть может функционировать без инфраструктуры (например, Bluetooth-соединения между устройствами). Если дополнить конечные узлы беспроводной сети функциями маршрутизации, то из таких комбинированных узлов может быть построена более сложная многошаговая сеть с ячеистой топологией, в которой задача выбора маршрутов и продвижения данных по сети решается распределенным образом с участием всех конечных узлов. Такая одноранговая сеть, состоящая из комбинированных хост-маршрутизаторов, также является сетью без инфраструктуры.

Подчеркнем, что отсутствие у сети инфраструктуры не означает отказ от организации скоординированной работы узлов этой сети. Это означает, что функции по координации конечных узлов выполняют сами эти узлы путем распределения работы между ними, а не заранее предустановленные специально предназначенные для этих целей инфраструктурные устройства.

Существуют два подхода к организации совместной работы узлов в безынфраструктурной сети. В первом случае эта функциональность равным образом распределена между всеми узлами сети, как это происходит, например, при маршрутизации, когда на каждом конечном

узле работает распределенный протокол маршрутизации OSPF. Другим подходом является делегирование функций по управлению сетью одному из конечных узлов. Например, для работы в качестве шлюза к проводной сети может быть динамически назначен один из конечных узлов беспроводной сети.

Самоорганизующиеся сети AdHoc

До недавнего времени типичным сценарием использования ноутбука, лэптопа или смартфона для сетевой работы было получение через интернет набора услуг, предоставляемых его инфраструктурой (включая маршрутизацию, доступ, например, к веб-серверу или к какому-либо удаленному сетевому приложению) и обмен данными с другими пользователями.

Однако в некоторых случаях люди, находящиеся достаточно близко друг от друга, например, спасатели в районе катастроф или военные на поле боя, могут предпочесть или быть вынуждены работать без какой-либо сетевой инфраструктуры, без доступа к интернету, только полагаясь на ресурсы своих мобильных вычислительных устройств. В этом случае их мобильные беспроводные вычислительные устройства должны уметь самостоятельно организовываться в сетевую структуру, полагаясь только на самих себя для установления и поддержания соединений (рис. 22.6).

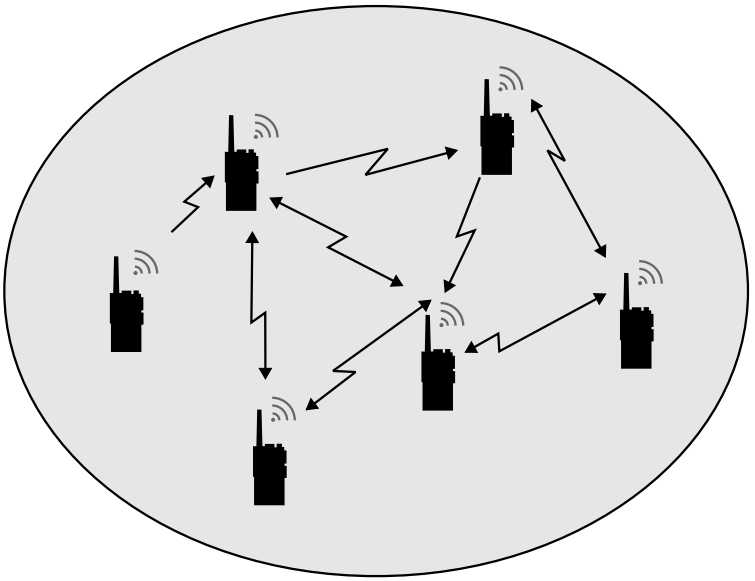


Рис. 22.6. Пример простейшей AdHoc-сети

Сети, состоящие из одноранговых (функционально равных) узлов, находящихся в зоне действия друг друга и взаимодействующих децентрализованным, самоорганизующимся образом без какой-либо центральной инфраструктуры и без заранее предопределенной топологии, называются **AdHoc-сетями**.

Термин AdHoc, в переводе с латинского означающий «для этой цели» или «по случаю», подчеркивает важное свойство таких сетей, а именно спонтанность образования сети, нацеленность ее на удовлетворение текущих, возможно, кратковременных нужд.

Как следует из определения, сеть AdHoc всегда является безынфраструктурной, но справедливо ли обратное?

Рассмотрим два сценария образования сети из одноранговых узлов. Каждый из них обладает всеми возможностями, которые в совокупности с возможностями других узлов позволяют сети нормально функционировать без инфраструктуры. Чтобы между узлами могла самопроизвольно установиться связь, они должны быть оснащены совместимыми сетевыми интерфейсами и соответствующими стеками протоколов, в которых заложена процедура автоматического присоединения к сети.

В первом сценарии узлы имеют сетевые интерфейсы, которые позволяют любому узлу самопроизвольно присоединяться к любому другому достижимому узлу. Такая случайно образованная сеть является AdHoc-сетью.

Во втором сценарии сетевые интерфейсы узлов ограничены установлением только таких связей, которые образуют сеть с предопределенным составом сети и предопределенной (не спонтанной!) топологией. Например, такой сетью будет сеть датчиков состояния окружающей среды, установленных в определенных точках леса. Эти датчики непосредственно или же через промежуточные узлы взаимодействуют друг с другом, образуя безынфраструктурную сеть, но это не AdHoc-сеть: она была заранее спланирована и ее узлы предварительно сконфигурированы.

Говоря здесь о сетях AdHoc, мы имеем в виду не сетевую технологию, а сетевую концепцию. Известно, что сетевая технология — это полный набор коммуникационных протоколов и реализующих их программных и аппаратных средств, стандартов, технической документации и других компонентов, необходимых и достаточных для создания сети и поддержания ее работы. Концепции ограничиваются набором основных принципов и идей, к числу которых для AdHoc-сетей относятся способность к самоорганизации, спонтанность сетевых соединений, особенности топологии и другие свойства, такие как:

- *Децентрализованная структура* — отсутствует централизованное управление, и все устройства участвуют равным образом в маршрутизации и управлении сетью.
- *Отсутствие инфраструктуры* — отсутствуют предустановленные маршрутизаторы или базовые станции, а каждый сетевой узел функционирует одновременно как хост и как маршрутизатор, тем самым образуя одноранговую структуру.
- *Динамическая топология* — AdHoc-сети характеризуются способностью адаптироваться к изменяющимся условиям. Устройства могут подключаться к сети и отключаться от нее в любое время, а сеть автоматически перестраивается, поддерживая новую конфигурацию соединений.
- *Многошаговые связи* между узлами — обмен данными в общем случае осуществляется через промежуточные узлы; AdHoc-сети могут иметь различные топологии, наиболее распространенные из которых — звездообразная, ячеистая и частично ячеистая.
- *Недостаточная масштабируемость* — при большом количестве узлов производительность сети падает из-за увеличения накладных расходов на маршрутизацию, перегрузки линий связи и интерференции.

Наиболее привлекательным свойством сетей AdHoc являются самозапускающиеся процедуры реконфигурирования сети, которые выполняются, когда какой-либо узел выходит из строя или появляется новый узел. Независимо от типа узла процедура его присоединения включает следующие шаги:

- оповещение-обнаружение соседей путем обмена маячковыми сообщениями или пассивного прослушивания среды;
- установление связности между узлами, включая измерение качества сигналов, возможную аутентификацию или обмен идентификаторами;
- автоматическое самоконфигурирование — присвоение IP-адреса или MAC-адреса;
- маршрутное присоединение — изучение маршрутов к другим узлам и объявление о своем присутствии.

Узлы сами обнаруживают соседей и самостоятельно связываются в сеть. (Хотя иногда допускается участие пользователя в настройке или установке режима работы узла на начальном этапе его подключения.)

Выше мы не раз по умолчанию предполагали, что AdHoc-сеть построена на основе беспроводных связей. Однако концепция AdHoc не требует, чтобы связи узлов были непременно беспроводными. Потребность в самоорганизующихся сетях может возникать и в тех случаях, когда узлы связываются кабелями. Проводная AdHoc-сеть может включать устройства, напрямую соединенные друг с другом с помощью кабелей Ethernet, USB или других физических соединений, без центрального коммутатора или маршрутизатора. Устройства должны самостоятельно назначать себе IP-адреса, например, с помощью локальной адресации или ручной настройки, использовать одноранговое взаимодействие и маршрутизацию между устройствами (при необходимости многошаговой передачи), возможно, реализовывать программную маршрутизацию. Классический Ethernet на коаксиальном кабеле (особенно оригинальные стандарты 10BASE5 и 10BASE2) функционально представлял собой сеть AdHoc, хотя термин «AdHoc» в то время не был широко распространен. Проводные AdHoc-сети технически возможны, а иногда и более предпочтительны, чем беспроводные, однако из-за большей гибкости беспроводная сеть является доминирующей формой AdHoc-сетей в большинстве практических приложений.

Минимальная конфигурация и возможность быстрого развертывания делают сети AdHoc удобными для решения экстренных краткосрочных задач, таких как военные сети на поле боя, ликвидация последствий стихийных бедствий и спасательные операции. Парадигма самоорганизующихся AdHoc-сетей положена в основу различных сетевых технологий и прежде всего мобильных сетей MANET, сетей беспилотных летающих аппаратов FANET и автомобильных сетей VANET, а также беспроводных сенсорных сетей (Wireless Sensor Network, WSN), в которых большое количество датчиков взаимодействуют друг с другом для сбора данных о параметрах окружающей среды, и др.

Мобильность беспроводных узлов

Мобильность не является неотъемлемым свойством беспроводных сетей. Существуют сети, состоящие из стационарных/фиксированных узлов, соединенных беспроводными связями. Например, два удаленных стационарных офиса, расположенных в труднодоступ-

ной местности, могут соединяться беспроводной спутниковой связью. Беспроводная связь применяется вместо проводной, когда по какой-то причине невозможно или невыгодно использовать кабельные линии связи. Причины могут быть разными: малонаселенная или труднодоступная местность; здания, имеющие историческую ценность, стены которых непозволительно подвергать испытанию прокладкой кабеля; необходимость организации временной связи, например, при проведении конференции в здании, в котором отсутствует проводной канал, имеющий достаточную для качественного обслуживания пропускную способность, и т. д. Таким образом, беспроводная связь не обязательно приводит к мобильности узлов. Однако чаще всего узлы беспроводных сетей являются мобильными.

Мы использовали выше термины «мобильность» и «мобильный» в широком, «бытовом» смысле, имея в виду просто возможность физического перемещения сетевых устройств с места на место. В контексте сетевых технологий термин «мобильность» используется более узко, в частности, не каждый узел сети, способный перемещаться между точками подключения, является мобильным. Например, с точки зрения пользователя, ноутбук, на котором он работает, перемещаясь из офиса в кафе или домой, вполне подходит под определение «мобильный», а для сети тот же ноутбук, подключающийся и отключающийся к/от нее во время перемещений пользователя, ничем не отличается от других, стационарных узлов этой сети.

В сетевых технологиях определены несколько различных типов мобильности, которые наряду с перемещением узла в пространстве учитывают и другие обстоятельства работы сети. Иногда не просто сказать, с каким типом мобильности мы имеем дело в той или иной ситуации. Является ли телефон, подключенный к Wi-Fi, мобильным устройством, когда он перемещается в пределах квартиры или здания? Или когда телефон со своим владельцем быстро едет от одной соты сети мобильной связи к другой, например в автомобиле? А если вы, закончив работу в офисе, закрываете ноутбук и едете домой, где продолжаете работу с того места, где остановились, является ли это устройство мобильным? Являются ли дроны, зависшие над поверхностью земли для трансляции связи с интернетом, мобильными узлами?

Возникает вопрос: как влияет перемещение устройства в пространстве на качество услуг, получаемых этим устройством от сети? Это влияние может быть заметным для пользователя, например, при удалении ноутбука от точки доступа Wi-Fi может ухудшиться качество приема или совсем прерваться связь. Или не заметным, например, качество разговора по мобильному телефону во время поездки на машине обычно не очень отличается от разговора из дома.

> Говорят, что сеть *поддерживает мобильность* устройства, если ее технические средства способны преодолеть проблемы {ухудшение функционирования системы или разрыв соединения с сетью), порожденные перемещением устройства. Технические средства системы должны предпринять те или иные действия , чтобы сгладить или полностью устранить эти последствия.

Значение термина «мобильный узел» изменяется от одной сетевой технологии к другой. Разные технологии по-разному купируют те или иные последствия того, что узел меняет свое местоположение. В одной технологии ставится задача не допустить прерывание сессий прикладного уровня при движении узла. Такими технологиями являются, например, сотовая связь или системы управления роями беспилотных летающих аппаратов. Для этих технологий узел рассматривается как мобильный, если для него необходимо все время поддерживать текущую сессию независимо от его перемещения.

В некоторых случаях для удобства пользователя, желающего сохранять при каждом повторном входе свой профиль — идентификатор, пароль, права доступа к файлам и устройствам, настройки пользовательской среды и др., — операционная система может поддерживать услугу логического входа, в соответствии с которой пользователь сети может подключаться к разным точкам доступа, каждый раз получая ту же самую знакомую рабочую среду, хотя ему может назначаться отличный от предыдущего входа IP-адрес. Такая услуга может рассматриваться как поддержание мобильности клиента. Другая технология — mobile IP[1] — нацелена на сохранение IP-адреса «блуждающего» по сети узла, это, например, дает возможность клиентам находить сервер, расположенный на этом узле, как бы последний ни перемещался. Для этой технологии мобильным узлом является узел, сохраняющий IP-адрес при перемещениях между точками доступа в большой сети.

Для сети MANET мобильность означает прежде всего требование поддержания работы сети при постоянно изменяющейся топологии сети из-за изменения положения узлов. Даже временно потерявший связь, отключившийся от сети MANET узел продолжает рассматриваться ею как мобильный, потому что в тот момент, когда он снова попадет в зону досягаемости других узлов сети, он автоматически (AdHoc) присоединится к сети и продолжит обмениваться данными с другими узлами сети.

Мобильность узла может обеспечиваться различными уровнями сетевых технологий.

Рассмотрим, например, беспроводную сеть Wi-Fi. Пользователь ходит с телефоном по своему дому. При его перемещениях качество сигнала между его телефоном и точкой доступа меняется (меняется мощность сигнала из-за расстояния до точки доступа, наличия стен и других препятствий, появления помех от микроволновой печи и пр.). Сеть и телефон экранируют эти изменения за счет настройки уровня мощности передаваемого сигнала, перехода на другой метод модуляции. Результат — физическое соединение между узлом и сетью не прерывается, сохраняя для пользователя телефона требуемые характеристики качества обслуживания. Телефон в этом примере является мобильным устройством для физического уровня сети, так как сетевые средства именно этого уровня участвуют в экранировании перемещений. В то же время протоколы верхнего уровня точки доступа и телефона не считают узел мобильным, так как его перемещения не потребовали от этих уровней никаких специальных действий.

Ситуация может быть иной: на физическом уровне мобильность не поддерживается, но она обеспечивается на транспортном или прикладном уровнях. Например, дрон в сети MANET может на время терять связь с остальными узлами (дронами) сети, и физический уровень не предпринимает никаких усилий по восстановлению связи, так как просто не может этого сделать. Но транспортный или прикладной протокол такой сети могут экранировать временную потерю связи с сетью за счет различных приемов, например, увеличения таймаутов транспортной сессии или периодического сохранения контекста приложения для продолжения его работы «с того же места» при восстановлении связи.

О безопасности беспроводных сетей

Все вопросы, которые мы кратко рассмотрим в этом разделе, подробно обсуждаются в части VII «Безопасность компьютерных сетей». Многое из того, что там написано, относится

[1] Об этой технологии читайте в разделе «Мобильный IPv4» главы 14.

как к проводным, так и беспроводным сетям. В то же время принципиальное различие этих двух видов сетей привнесло свою специфику в набор уязвимостей, угроз и методов противодействия им.

В проводных сетях, в которых данные передаются по витой паре или по оптоволоконным кабелям, злоумышленникам для воплощения их планов необходимо получить физический доступ к порту коммутатора или маршрутизатора. Проводные атаки обычно требуют физического проникновения, взлома замка или вмешательства инсайдера, что снижает вероятность внешних угроз.

Беспроводные сигналы передаются посредством радиоволн, которые распространяются за пределы физических стен. Беспроводная среда передачи по своей природе более открыта для посторонних. Она не может быть физически ограничена, а значит, злоумышленник имеет возможность перехватывать данные, просто разместив приемное устройство в любом месте в пределах зоны передачи отправителя. Злоумышленникам не нужно входить в здание, где размещена сеть вашей организации. Им достаточно находиться рядом с этим зданием в зоне покрытия сети, например в припаркованном автомобиле с чувствительной антенной. Отсюда следует, что *беспроводные сети более уязвимы к внешним угрозам, чем проводные*.

Перехват незашифрованных пакетов возможен с помощью общедоступных устройств и программ, таких как Wi-Fi-устройства в режиме мониторинга, приложение захвата и анализа трафика Wireshark, любой другой сетевой сканер или просто смартфон. Из-за легкости прослушивания трафика *шифрование в беспроводной сети является обязательным*, и это еще одно важное отличие беспроводной сети от проводной, для которой шифрование может быть опциональным.

В беспроводной сети обязательной является и процедура логического присоединения к сети, которая обычно отсутствует при физическом подключении устройства к проводной сети. Процедура присоединения к беспроводной сети обязательно включает выработку ключей шифрования и аутентификацию пользователя. *Аутентификация* — это процедура доказательства субъектом/объектом того, что он есть то, за что (кого) себя выдает. Таким образом, обеспечение безопасности беспроводной сети начинается до подключения к среде, а проводной — после подключения.

В число уязвимостей беспроводной сети входит *подмена легитимной точки доступа к сети* — спуфинг точки доступа. Злоумышленник создает беспроводную точку доступа, которая выдает себя за настоящую, легальную беспроводную сеть, чтобы обманным путем заставить пользователей или устройства подключиться к этой фальшивой сети, а затем манипулировать сетевым трафиком, например, прочитать его, если он не зашифрован. Таким путем могут также быть получены учетные данные и пароли.

Для обмана злоумышленники с помощью вредоносного программного обеспечения временно превращают свой ноутбук или телефон в поддельную точку доступа, которая передает в эфир имя сети, похожее на имя настоящей сети. Чтобы пользователи могли автоматически подключиться, поддельная точка использует более сильный уровень сигнала или позиционирует себя ближе к пользователям. Главной же причиной, которая делает возможной подмену сети, является отсутствие взаимной аутентификации: в старых средствах безопасности беспроводных сетей клиент проходит аутентификацию у сети, а сеть не проходит обратную аутентификацию у клиента. Это создает «одностороннее доверие»: пользователь доверяет сети, а сеть не подтверждает свою легитимность. Для защиты от

такого эффекта используются более совершенные способы взаимной аутентификации, в которых сеть и пользователь аутентифицируют друг друга.

Специфической атакой на беспроводную сеть является глушение. В проводных сетях такой проблемы нет, поскольку сигналы передаются по кабелям.

Глушение беспроводного сигнала — это противозаконное создание помех в радиоканале с целью нарушить или ухудшить нормальное соединение беспроводных устройств. В отличие от атак на шифрование или пароли, направленных на нарушение конфиденциальности и целостности информации, глушение имеет целью нарушить *доступность* сети. Поскольку сигналы распространяются по воздуху, любой близлежащий передатчик может создавать помехи. Однако обычно для глушения используются специальные устройства, называемые генераторами помех или подавителями сигнала. Генераторы помех работают в определенном радиусе на частоте устройства, работе которого нужно помешать. Эффект от глушения очень похож на эффект от DDoS-атаки: сеть становится недоступной или труднодоступной.

Ограничения энергопотребления

При изучении проводных сетей мы практически не затрагивали вопросы энергоэффективности протоколов, технологий и узлов сети. И это понятно: конечные и коммуникационные узлы таких сетей питаются, как правило, от сети, поэтому при разворачивании проводной сети достаточно было просто позаботиться о наличии необходимого количества розеток. Вопрос об энергопотреблении узлов проводной сети встает только при планировании емкости резервных источников питания UPC на случай аварий в энергосетях.

В мире беспроводных сетей картина совершенно иная. Ограниченное энергопотребление является одним из их самых важных отличительных свойств. Практически все устройства беспроводной сети — смартфоны, ноутбуки, датчики, исполнительные механизмы автоматизированных промышленных и домашних систем, дроны и другие узлы интернета вещей — работают от батарей, а значит, энергосбережение становится критически важной задачей для поддержания работоспособности беспроводной сети. Главная цель энергосбережения — максимально продлить время работы сети при сохранении надежной связи между узлами.

Каждый из нас очень часто решал эту задачу, заботясь о продлении жизни своего смартфона, и прибегал к разным приемам: выключая его на время, уменьшая яркость экрана, беря с собой пауэрбанк и т. п. Примерно такие же приемы используются и в беспроводных сетях, но на системной основе.

Не все типы беспроводных сетей в равной степени заботятся об экономии энергии своих узлов. Требования к энергоэффективности для разных типов беспроводных сетей сильно разнятся в зависимости от их архитектуры, мобильности, ограничений устройств и условий эксплуатации.

Очевидно, что в сети датчиков, установленных вдоль магистрального газопровода, проходящего через малонаселенные или просто пустынные местности, очень желательно, чтобы датчики могли работать на одной батарее без подзаряда месяцы и даже годы. Для таких сетей разрабатываются технологии, обеспечивающие работу узла на одной маленькой пуговичной батарее в течение 10 лет.

Спутниковые сети также создаются с учетом проблем энергосбережения, но в этих сетях проблемы решаются в другой шкале энергопотребления. Передатчики геостационарных спутников расходуют на передачу данных на расстояние 36 000 км сотни ватт. Здесь часто используются возобновляемые источники энергии — солнечные батареи.

Сети дронов являются одними из самых чувствительных к энергозатратам типов беспроводных сетей. Эффективность энергопотребления дрона напрямую увеличивает время полета и продолжительность миссии. Например, типичный небольшой дрон может иметь аккумулятор емкостью 5200 мАч. Такой дрон потребляет около 12 А от своей батареи, что обеспечивает ему время полета всего около 25 минут. Бóльшая часть электроэнергии дрона тратится на полет. Но затратами на работу коммуникационной системы тоже нельзя пренебрегать. Каждый переданный дроном бит увеличивает нагрузку на его энергосистему, поэтому очень важно минимизировать энергетические затраты на передачу данных между дронами, на основе которой строится координация роя дронов, решаются задачи навигации и маршрутизации. Вычислительные задачи, решаемые процессором дрона, также истощают его батарею, поэтому вместо выполнения сложных вычислений на борту необработанные данные передаются для обработки на наземные станции.

Менее жесткие требования к эффективности энергопотребления предъявляют сотовые мобильные сети и сети Wi-Fi, у которых за конечными устройствами есть постоянный присмотр со стороны пользователя, который вовремя их подзаряжает, а точки доступа Wi-Fi и базовые станции сотовых сетей питаются от центральной электрической сети. Это, однако, не означает, что разработчики данного типа сетей не занимаются проблемой продления «энергетической» жизни пользовательских устройств.

Для экономии электроэнергии в беспроводных сетях используются различные приемы. Прежде всего, все коммуникационные протоколы беспроводных сетей проектируются с учетом энергии, которая тратится узлом при реализации этого протокола — аспект, который разработчики протоколов проводных сетей обычно не принимают во внимание.

Одним из основных факторов, влияющих на интенсивность потребления энергии узлом сети, является степень его активности. Например, номинальное потребление типичного интерфейса Wi-Fi 802.11n в режиме с одной антенной составляет 1280 мА / 940 мА / 820 мА /100 мА в режимах передачи/приема/ожидания/спящего режима соответственно. Отсюда становится понятной задача разработчика протокола: он должен продумывать алгоритм взаимодействия устройств таким образом, чтобы они проводили бóльшую часть своего жизненного цикла в спящем состоянии.

Узлы должны регулировать мощность радиосигнала в зависимости от расстояния до следующего узла. Мощность передатчика должна быть достаточной, чтобы не увеличивать число повторных передач, но не больше, чем необходимо для установления удовлетворительного соединения между двумя узлами сети. Таким образом исключаются потери энергии из-за постоянного использования избыточной мощности передачи.

В беспроводных сетях даже протоколы маршрутизации разрабатываются с учетом энергопотребления. Поясним это на примере мобильной AdHoc-сети (MANET), показанной на рис. 22.7. Пусть спасательная операция проводится группой людей, каждый из которых оснащен узлом беспроводной связи — рацией, которая включает в себя также функции маршрутизатора. Эти узлы объединены в сеть по технологии AdHoc, так что сообщение может передаваться через промежуточные узлы по некоторому маршруту, выбираемому динамически протоколом маршрутизации. Мобильность раций, которые перемещаются

со своими владельцами, приводит к тому, что постоянно меняется граф связей и протокол маршрутизации постоянно ищет новые маршруты. Возникает вопрос: на каких принципах должна быть основана маршрутизация, чтобы рации сохраняли свой заряд как можно дольше и чтобы команда спасателей как можно дольше работала в полном составе?

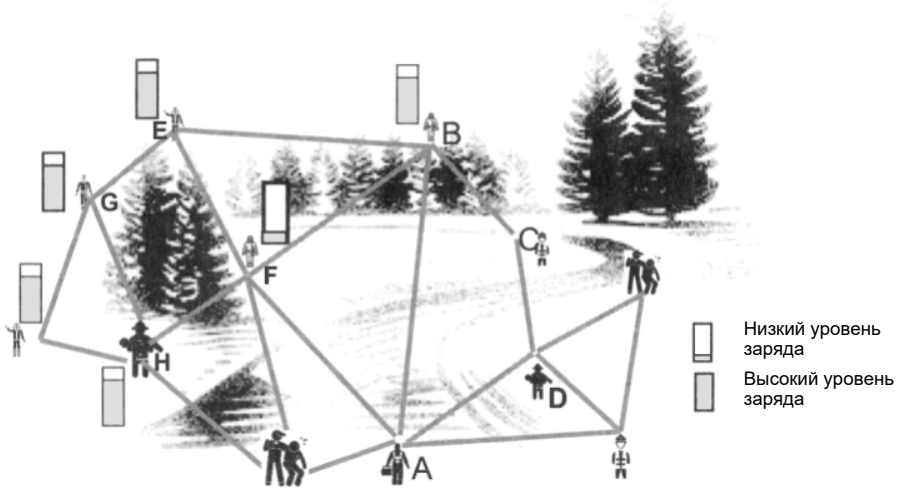


Рис. 22.7. Мобильная AdHoc-сеть MANET, образованная командой спасателей

Одним из принципов могла бы быть минимизация расстояния одного шага передачи. Передача данных на большие расстояния потребляет больше энергии, так как передатчик должен излучать сигнал большей мощности. Поэтому протокол маршрутизации должен выбирать более короткие пути или многошаговую передачу для снижения энергопотребления. Пусть, например, спасатель *A* решает послать сообщение своему коллеге *B*, находящемуся от него на большом расстоянии. Энергосберегающая система маршрутизации в таком случае должна предпочесть маршрут, состоящий из нескольких промежуточных шагов — через спасателей *D* и *C*, находящихся на небольших расстояниях друг от друга, а не прямую передачу удаленному члену команды. Каждый транзитный участок потребляет меньше энергии, что ведет к общей экономии энергии передатчика. Как видим, такой подход полностью противоположен работе традиционных протоколов проводных сетей, которые обычно выбирают маршрут с минимальным количеством хопов.

Другой принцип — это выбор маршрутов с учетом уровня заряда узлов. Одной из стратегий энергосбережения является поддержание заряда рации у всех спасателей примерно равным, так как это предотвращает преждевременный выход из строя слабых в энергетическом смысле узлов и сохраняет максимальную численность активных спасателей. Если у кого-либо из команды заряд рации становится заметно меньше, чем у других (на рисунке — узел *F*), протокол маршрутизации освобождает его от промежуточных передач, тем самым помогая ему выровнять заряд с другими. То есть для передачи данных от спасателя *H* спасателю *B* энергосберегающий протокол маршрутизации выберет маршрут *H-G-E-B*, а не более короткий *H-F-B*.

Иногда для выживания сети протокол маршрутизации совсем исключает из рассмотрения узлы с маленьким зарядом. Узлы с более высоким остаточным зарядом имеют приоритет при пересылке пакетов. Для того чтобы этот подход работал, протокол маршрутизации должен иметь данные об оставшемся заряде батареи у каждого узла.

Экономия энергии также достигается разбиением всей группы спасателей на несколько подгрупп с назначением лидера в каждой подгруппе, так чтобы вся информация от членов подгруппы стекалась к лидеру, который потом передает ее в центр или делится ею с другими лидерами. Такой подход экономит энергию, так как лидер агрегирует данные членов своей группы и тем самым уменьшает общее число передач.

Типы беспроводных сетей и радиотехнологий

Выше мы определили ряд свойств, которые могут быть положены в основу разделения беспроводных сетей на классы: по зоне покрытия, по типу архитектуры (в зависимости от наличия или отсутствия предустановленной инфраструктуры), по степени мобильности, по виду используемой радиотехнологии, по области применения. Все эти типы сетей будут с разной степенью подробности рассмотрены в следующих главах.

Начнем наш обзор с классификации беспроводных сетей в зависимости от *зоны покрытия* — то есть области пространства, в которой сигнал от узлов беспроводной сети достаточно силен для того, чтобы устройства, расположенные в этой области, могли устанавливать надежные соединения и получать доступ к этой сети. Зона покрытия беспроводной сети аналогична территории покрытия проводной сети, поэтому здесь так же, как в проводных сетях, есть локальные, городские и глобальные сети. Но начнем мы с самых компактных — так называемых носимых и персональных беспроводных сетей.

- **WBAN** (Wireless Body Area Network) — **беспроводная носимая сеть** — это тип беспроводной сети, предназначенный для соединения различных медицинских датчиков и устройств, размещаемых на теле человека или вокруг него.
- **WPAN** (Wireless Personal Area Network) — **беспроводная персональная сеть** — это тип сети, соединяющей устройства, такие как смартфоны, компьютеры и ноутбуки, обычно в радиусе нескольких метров. WPAN в основном используют протоколы беспроводной связи ближнего радиуса действия, такие как Bluetooth, ZigBee, Thread, и часто применяются для персональных устройств, расположенных в одной комнате или на рабочем месте.
- **WLAN** (Wireless Local Area Network) — это **беспроводная локальная сеть**, дальность действия которой составляет 30–100 метров, позволяющая устройствам подключаться и обмениваться данными в пределах определенной зоны, например, дома или офиса. К такому типу сетей относятся сети Wi-Fi (IEEE 802.11).
- **WMAN** (Wireless Metropolitan Area Network) — это **беспроводная городская сеть**, предназначенная для покрытия расстояния до 50 километров.
- **WWAN** (Wireless Wide Area Network) — это **беспроводная глобальная сеть**, обеспечивающая беспроводное соединение на большой территории, охватывая регионы, города,

страны и даже весь мир. К таким сетям относятся сотовые (4G/LTE, 5G) и спутниковые сети.

Следующий критерий классификации — *наличие инфраструктуры* — в основном применяется для беспроводных сетей; в зависимости от него сети делятся на классы:

- **Инфраструктурные сети**, в состав которых обязательно входят инфраструктурные узлы, такие как точки доступа/базовые станции. Пример: Wi-Fi с точками доступа, сотовые сети, спутниковые сети.
- **Сети AdHoc**, в которых инфраструктура отсутствует. Все узлы такой сети функционально равны и, находясь в зоне действия друг друга, взаимодействуют децентрализованным, самоорганизующимся образом без какой-либо центральной инфраструктуры. К таким сетям относятся следующие типы сетей:
  - **MANET** (Mobile AdHoc Network) — **мобильная сеть AdHoc**, часто состоящая только из мобильных портативных устройств, подобных смартфонам, ноутбукам, лэптопам, так чтобы они могли взаимодействовать друг с другом прямо, не полагаясь на какую-либо инфраструктуру.
  - **VANET** (Vehicular AdHoc Network) — **транспортная сеть AdHoc**, предназначенная для использования в транспортных системах для обеспечения связи между автомобилями и придорожной инфраструктурой в целях безопасности и удобства водителей, пассажиров и пешеходов.
  - **FANET** (Flying AdHoc Network) — **летающая сеть AdHoc**, или **сеть дронов**, предназначенная для поддержки связи между автономными транспортными средствами, способными летать (дронами, БПЛА), и рядом сетевых объектов, развернутых на земле.
  - **WSN** (Wireless Sensor Network) — **беспроводная сенсорная сеть**, представляющая собой совокупность пространственно распределенных сенсорных узлов, которые отслеживают параметры среды, такие как температура, звук или загрязнение, и передают данные по беспроводной связи на центральный узел-приемник для обработки. Датчики путем самоорганизации образуют многошаговую сеть.
- **Гибридные сети**. Как всегда, стройную таксономию нарушают гибридные объекты, то есть объекты, которые могут быть отнесены сразу к нескольким классам. В данном случае это сети AdHoc, в которых многошаговая сеть Wi-Fi с ячеистой топологией сочетается с наличием инфраструктуры. Подробнее о этом читайте в разделе «Ячеистая архитектура сети Wi-Fi» главы 25.

В зависимости от *степени мобильности* сети делятся на:

- **Стационарные сети**, в которых пользовательские устройства стационарны, то есть если они и перемещаются, то только в небольшом диапазоне. Пример: статические сенсорные сети — LoRaWAN, спутниковая сеть Starlink (спутники, естественно, летают, а вот устройства пользователя должны быть стационарны).
- **Маломобильные сети**, связывающие между собой смартфоны, ноутбуки и другие устройства интернета вещей, например сети ZigBee и Wi-Fi.
- **Высокомобильные сети**, узлами которых являются транспортные средства (сети VANET), поезда, БПЛА (сети FANET). Например, сотовые сети 5G обеспечивают связь устройств пользователей, движущихся со скоростью до 500 км/ч.

Часто тип сети обозначают названием *технологии*, лежащей в ее основе, например «сеть **4**G», «сеть Starlink»:

- Сети **Wi-Fi** (IEEE 802.11).
- Сети **Bluetooth** (802.15.1).
- Сети **ZigBee** (802.15.**4**) и сети **Thread**.
- Сети сотовой связи **4G LTE**, **5G NR**, **6G** (в стадии исследования).
- Сети спутниковой связи, системы геостационарных **(GEO)**, среднеорбитальных **(MEO)** и низкоорбитальных **(LEO)** спутников **(Iridium**, **Inmarsat**, **GPS**, **Starlink**, **OneWeb)**.
- Глобальные сети низкого энергопотребления **LoRaWAN**, **NB-IoT**.

И наконец, мы переходим к классификации беспроводных сетей по *областям применения*. Как и можно было ожидать, здесь мы сталкиваемся с огромным многообразием типов сетей.

Даже если бы мы ограничились только технологией Wi-Fi, то получили бы длинный список применений: персональные и домашние сети, локальные сети предприятий, Wi-Fi для пассажиров самолетов, сети госпиталей и больниц, тактические военные сети MANET, публичные точки доступа Wi-Fi, муниципальные сети, сети дронов и т. д.

Некоторые типы сетей и технологий настолько расширили свое присутствие в жизни человека, что трудно найти область, в которой они бы не применялись. Это касается прежде всего глобальной сети интернет. Действительно, кто возьмет на себя смелость перечислить области использования интернета? И какие типы устройств и аппаратных платформ, сетевых технологий и технологий безопасности включает интернет? К такого рода техническим **экосистемам** относятся также сотовые и спутниковые сети, которые обеспечивают разнообразные услуги публичного вещания, общественную и мультимедийную связь, доступ к интернету со спутников LEO, MEO, GEO. Они используются в авиации, морской и аэрокосмической отрасли и в бесчисленном множестве других областей. В последние годы такой вездесущей концепцией-экосистемой стал интернет вещей.

Интернет вещей (Internet of Things, **IoT**) — это сеть физических объектов («вещей»), оснащенных датчиками, исполнительными механизмами, встроенными контроллерами и коммуникационными средствами. Эта сеть позволяет *в реальном времени* передавать, собирать и анализировать данные, полученные от своих узлов. На основе полученных данных приложение интернета вещей может автоматически (или с участием человека) вырабатывать управляющие воздействия и передавать их по удаленной связи на исполнительные механизмы, например включать/выключать приборы. Часто датчики IoT собирают данные о состоянии физической среды (температуру, движение, влажность, освещенность) или информацию о состоянии устройств (включено/выключено, заряжено/разряжено, готово к работе/не готово).

Термин «интернет вещей» может вводить в заблуждение. Узлам IoT требуется сетевое подключение, но это не обязательно должно быть интернет-соединение. Для обмена данными узлы интернета вещей могут подключаться к сети *локально* (без интернета), например, устройства умного дома, подключенные только к локальному ZigBee-хабу, или медицинские устройства, соединенные в локальную сеть больницы. Другим вариантом подключения устройств IoT является *косвенный* доступ в интернет, через шлюз, роль которого, например, может играть смартфон. И наконец, интернет вещей может поддерживать *прямое* подключение устройств к интернету, обычно через Wi-Fi, NB-IoT или через сеть 4G/5G. Примером такого рода «вещей», непосредственно связанных с интернетом, явля-

ются умные видеокамеры (в домашних охранных системах), GPS-трекеры, показывающие в реальном времени на вашем мобильном телефоне перемещения вашего кота, или другие, более важные умные устройства.

Масштаб интернета вещей простирается от отдельных умных вещей, используемых людьми в повседневной жизни (браслет Fitbit, управляемый замок или тот же трекер для кота), до **Массового интернета вещей** (Massive Internet of Things, MIoT) — такого вида интернета вещей, который связан с развертыванием огромного количества (десятков тысяч или миллионов) недорогих, маломощных и низкоскоростных устройств с батарейным питанием, регулярно обменивающихся небольшими объемами данных по беспроводным сетям большой зоны покрытия. Приведем несколько областей использования Массового интернета вещей:

- мониторинг окружающей среды (датчики землетрясений/цунами, детекторы лесных пожаров, устройства для мониторинга качества воды, воздуха);
- удаленный мониторинг состояния промышленного оборудования, датчики состояния трубопроводов, умное электроснабжение, водоснабжение и газоснабжение;
- отслеживание диких животных/домашнего скота.

Для Массового интернета вещей характерно использование технологий низкого энергопотребления (им посвящена глава 26). Кроме того, Массовый интернет вещей является одной из трех основных категорий услуг сетей 5G, об этом также написано в главе 24.

Наряду с областью применения беспроводной сети того или иного типа, мы часто указывали радиотехнологию, которая преимущественно используется для ее построения. Понятно, что одна и та же технология может быть положена в основу разных типов сетей, отличающихся областями использования, а сети одного назначения могут использовать несколько разных радиотехнологий. В табл. 22.1 приводится соответствие (иногда довольно условное) между областями применения беспроводных сетей и доминирующими в этих нишах радиотехнологиями[1].

Таблица 22.1. Области применения радиотехнологий

	Wi-Fi	Bluetooth	ZigBee	Thread	LTE/5G	LoRaWAN	Спутник
Персональные сети (WPAN)	+	+					
Домашняя сеть	+		+	+			
Корпоративная сеть	+				+		
Публичная сеть					+		+
Военная сеть	+				+		+
Здравоохранение	+	+	+	+	+		
Интернет вещей	+	+	+	+	+	+	+
Промышленная связь					+	+	

[1] Все перечисленные в таблице технологии рассматриваются далее в этой части книги.

ГЛАВА 23 Технологии физического уровня беспроводных сетей

Элементы беспроводной линии связи

Структура беспроводной линии связи во многом похожа на структуру проводной линии связи, рассмотренной ранее в главе 6. Основным ее отличием (рис. 23.1) является использование антенны — устройства, преобразующего колебания электрического тока в проводе в колебания электромагнитных волн, излучаемых в окружающую среду. Антенна также выполняет обратное преобразование: она улавливает колебания электромагнитного поля в окружающей среде и преобразует их в колебания электрического тока.

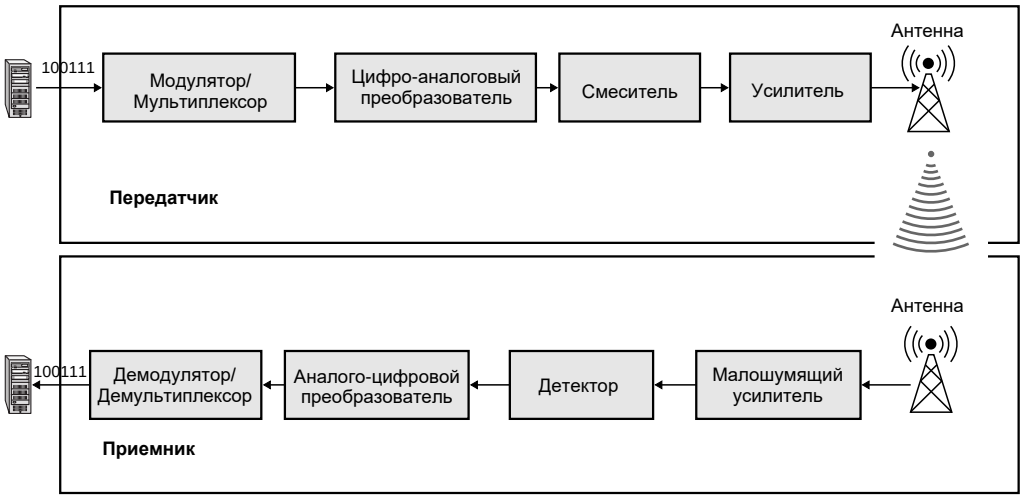


Рис. 23.1. Структура беспроводной линии связи

Типы антенн и их устройство подробно рассматриваются далее в отдельном разделе.

В состав передатчика кроме антенны также входят:

- **Модулятор.** Этот блок принимает на вход двоичные данные от источника данных (DTE) и заменяет один или порцию бит советующим символом кода. Наиболее часто применяемыми кодами в радиоканалах являются коды BPSK, QPSK и коды семейства QAM, от 16-QAM до 1024-QAM. Для их генерации применяются различные варианты квадратурного модулятора, рассмотренного в главе 7, от полностью аналогового до

полностью цифрового (выполняющего прямой цифровой синтез символом кода). На рисунке показан вариант цифрового модулятора, после которого требуется цифро-аналоговое преобразование.

- **Мультиплексоры.** В беспроводных линиях применяются различные виды мультиплексирования, частотное (FDM), временно́е (TDM), их комбинации, а также специфические для беспроводных сетей виды, позволяющие распределить спектр модулированного сигнала по всей доступной полосе частот в целях уменьшения влияния помех на полезные сигналы, они рассматриваются далее в разделе «Техника расширенного спектра». На рисунке блок мультиплексора показан совмещенным с блоком модулятора, так как при частотном мультиплексировании эти две функции часто выполняются в рамках одной процедуры.
- **Цифро-аналоговый преобразователь** (ЦАП) принимает дискретную последовательность значений модулирующего сигнала и преобразует ее в плавный аналоговый сигнал.
- **Смеситель.** Этот блок выполняет смешение модулирующего сигнала (его синфазной и квадратурной составляющих) с высокочастотным несущим сигналом, на его выходе получается модулированный высокочастотный сигнал. Как было сказано в главе 7, смеситель может быть как аналоговым, так и цифровым. На рисунке выбран вариант аналогового смесителя, поэтому он помещен после ЦАП.
- **Усилитель**. Дальность беспроводной линии связи зависит от мощности передаваемого в окружающую среду радиосигнала, для этого выходной сигнал смесителя необходимо усилить.

В состав приемника входят:

- **Малошумящий усилитель.** На вход приемника поступает сигнал небольшой мощности, ослабленный при распространении в воздухе, а возможно, и в более плотных средах (стены и т. п.). Поэтому его нужно усилить, но не вносить дополнительные искажения из-за собственного шума усилителя.
- **Детектор.** Этот блок выделяет модулирующий сигнал, очищая общий сигнал от высокочастотной несущей — операция, обратная операции смесителя.
- **Аналого-цифровой преобразователь** (АЦП) преобразует модулирующий сигнал в цифровую форму, так как в нашем примере остальные блоки приемника обрабатывают сигналы в цифровой форме.
- **Блок демультиплексора/демодулятора** выполняет демультиплексирование различных частот в общем сигнале (если применялось FDM-мультиплексирование) и распознает символы примененного модулятором кода (например, 16-QAM) в демультиплекированном сигнале. Для этой операции обычно используется быстрое преобразование Фурье (FTT). Результатом работы этого комбинированного блока является последовательность бит исходных данных, поступивших на вход беспроводной линии связи.

Мы описали состав беспроводной линии связи очень обобщенно и упрощенно. В реальной линии связи могут применяться и блоки других типов, например, различные фильтры для придания спектру сигнала необходимой формы. Распределение функций между цифровой и аналоговой частями приемника и передатчика также может быть различным.

В рассмотренной беспроводной линии связи подразумевалось, что приемник и передатчик принадлежат к равноправным узлам сети, то есть сеть является одноранговой. В инфраструктурных беспроводных сетях, таких как сети 4G/5G и Wi-Fi с точкой доступа, линия

связи связывает два неравноправных узла — центральный (базовая станция, точка доступа) и клиентский (станция Wi-Fi, устройство пользователя сотовой сети). В таких сетях различают два режима передачи данных по беспроводной линии связи — **нисходящий**, от базовой станции 4G/5G или точки доступа Wi-Fi к устройствам пользователей (**DownLink, DL**), и **восходящий**, от устройств пользователей к базовой станции или точке доступа (**UpLink, UL**). При изучении технологий сетей с инфраструктурой эта терминология широко используется, так как часто в направления DL и UL применяются разные технические приемы и методы, это объясняется тем, что центральный элемент сети обладает более мощным процессором и устройством питания большей емкости.

Диапазоны электромагнитного спектра

Характеристики беспроводной линии связи — расстояние между узлами, территория охвата, скорость передачи информации и т. п. — во многом зависят от частоты используемого электромагнитного сигнала. На рис. 23.2 показаны диапазоны электромагнитного спектра. Обобщая, можно сказать, что они и соответствующие им беспроводные системы передачи информации делятся на четыре группы.

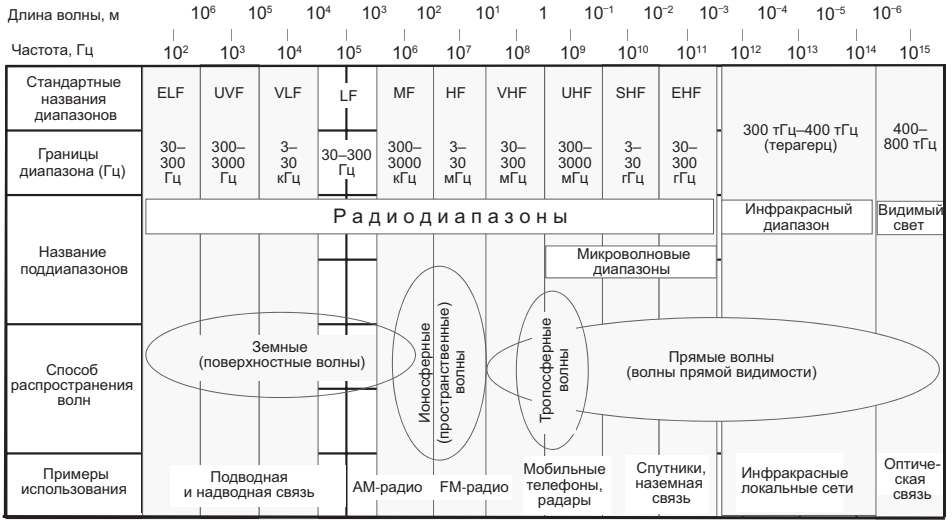


Рис. 23.2. Диапазоны электромагнитного спектра. Стандартные диапазоны частот: ELF — крайне низкие (КНЧ), UVF — ультранизкие (УНЧ), VLF — очень низкие (ОНЧ), LF — низкие (НЧ), MF — средние (СЧ), HF — высокие (ВЧ), VHF — очень высокие (ОВЧ), UHF — ультравысокие (УВЧ), SHF — сверхвысокие (СВЧ), EHF — крайне высокие (КВЧ)

Диапазон до 300 ГГц имеет общее стандартное название — радиодиапазон. Это очень широкий диапазон, поэтому организация ITU разделила его на несколько поддиапазонов (они показаны на рисунке), начиная от сверхнизких частот (Extremely Low Frequency, ELF) и заканчивая сверхвысокими (Extra High Frequency, EHF). Привычные для нас широковещательные радиостанции работают в диапазоне от 20 кГц до 300 МГц, и для этих диапазонов существует хотя и не определенное в стандартах, однако часто используемое название «широковещательное радио». В этом диапазоне работают низкоскоростные

радиомодемы, которые могут передавать двоичные данные по беспроводной линии связи со скоростями от 2400 бит/с до 19200 бит/с.

Несколько диапазонов от 300 МГц до 300 ГГц имеют также нестандартное название микроволновых диапазонов. Микроволновые системы представляют наиболее широкий класс систем, объединяющий радиорелейные линии связи, радары, спутниковые каналы, беспроводные локальные сети Wi-Fi и мобильные сотовые сети 4G/5G. Нужно отметить, что существуют различные трактовки границ микроволнового диапазона. Нижняя граница в 300 МГц сегодня часто считается устаревшей, пришедшей из мира радаров Второй мировой войны. Более современная точка зрения состоит в том, что микроволновый диапазон имеет границы 1 ГГц — 30 ГГц, а волны частоты выше 30 ГГц называют не микроволновыми, а миллиметровыми (mmWaves), это название стало особенно часто использоваться в инженерной литературе после появления сотовых сетей 5G, включивших миллиметровые волны в один из своих рабочих диапазонов.

В микроволновом и миллиметровом диапазонах работают также спутниковые беспроводые системы связи. Для них выделено несколько диапазонов: **L** (1–2 ГГц), **S** (2–4 ГГц), **C** (4–8 ГГц), **Ku** (12–18 ГГц), **Ka** (26,5–40 ГГц) и **V** (50–80 ГГц).

Выше микроволновых диапазонов располагается инфракрасный диапазон. Микроволновые и инфракрасный диапазоны также широко используются для беспроводной передачи информации. Так как инфракрасное излучение не может проникать через стены, системы инфракрасных волн служат для образования небольших сегментов локальных сетей в пределах одного помещения.

В последние годы видимый свет тоже стал применяться для передачи информации (с помощью лазеров). Системы видимого света используются как высокоскоростная альтернатива микроволновым двухточечным каналам для организации доступа на небольших расстояниях.

Мы завершим рассмотрение различных диапазонов частот, используемых в беспроводных линиях связи, анализом зависимости между *частотой несущей и битовой скоростью передачи двоичных данных с использованием этой несущей*. Хотя интуитивно кажется, что для достижения более высокой скорости передачи данных нужно использовать более высокие частоты несущей, формально формулы Шеннона и Найквиста об этом не говорят. Они говорят о том, что максимальная скорость передачи данных пропорциональна *ширине* полосы пропускания, отведенной для передачи модулированного сигнала, но ничего не говорят о том, как это связано со значением несущей. То есть теоретически можно предположить, что если для более низкой несущей выбрать полосу шире, чем для более высокой несущей, то скорость передачи данных на низкочастотной несущей будет выше, чем на высокочастотной. На практике это не так, и мы в следующих разделах при изучении различных технологий беспроводных сетей, таких как Wi-Fi и технологий сотовых мобильных сетей 4G/5G, увидим, что значительное повышение скорости передачи данных происходит за счет освоения новых, более высоких диапазонов частот, например, диапазона 6 ГГц технологией Wi-Fi (до этого она использовала только диапазоны 2,4 ГГц и 5 ГГц) или диапазона 24–71 ГГц технологией 5G (ее предшественница, технология 4G, работала только в диапазонах до 2,6 ГГц).

Инженерная практика построения радиоцепей показала, что очень сложно построить такой радиопередатчик, который бы позволял передавать частоты, значительно отклоняющиеся от несущей частоты. Все элементы беспроводной линии связи — фильтры, антенны, сме-

сители, детекторы — проектируются для передачи определенной частоты, и значительные отклонения от нее реализовать очень сложно или же невозможно. Поэтому в инженерной практике построения радиоцепей имеется такое понятие, как **относительная (фракционная) полоса пропускания B**. Это та величина полосы пропускания, на которую можно отклониться от значения несущей частоты при проектировании радиоцепей. Она равна:

$$B = k \times fc,$$

где B — относительная полоса пропускания;

fc — значение несущей частоты;

k — отношение (фракция) полосы пропускания к значению несущей частоты.

Значение k не превышает 0,05 (5 %), поэтому полоса пропускания у сигнала с несущей 20 ГГц может быть равной 0,05 × 20 = 1 ГГц, а у сигнала с несущей 2,4 ГГц — только 120 МГц. Соответственно, максимальная скорость передачи данных (при одинаковом способе модуляции сигнала) будет отличаться почти в 10 раз.

Распространение электромагнитных волн

Прежде всего вспомним о нескольких важных физических явлениях, связанных с распространением волн вообще и электромагнитных волн в частности. На рис. 23.3 показано, что сигнал, встретившись с препятствием, может распространяться в соответствии с тремя механизмами: отражением, дифракцией и рассеянием. Когда сигнал встречается с препятствием, частично прозрачным для данной длины волны и в то же время имеющим размеры, намного превышающие длину волны, часть энергии сигнала **отражается** от этого препятствия. Если сигнал встречает непроницаемое для него препятствие (например, металлическую пластину) намного большего размера, чем длина волны, то происходит **дифракция** — препятствие как бы огибается сигналом, что позволяет получить его, даже не находясь в зоне прямой видимости. И наконец, при встрече с препятствием, размеры которого соизмеримы с длиной волны, сигнал **рассеивается**, распространяясь под различными углами.

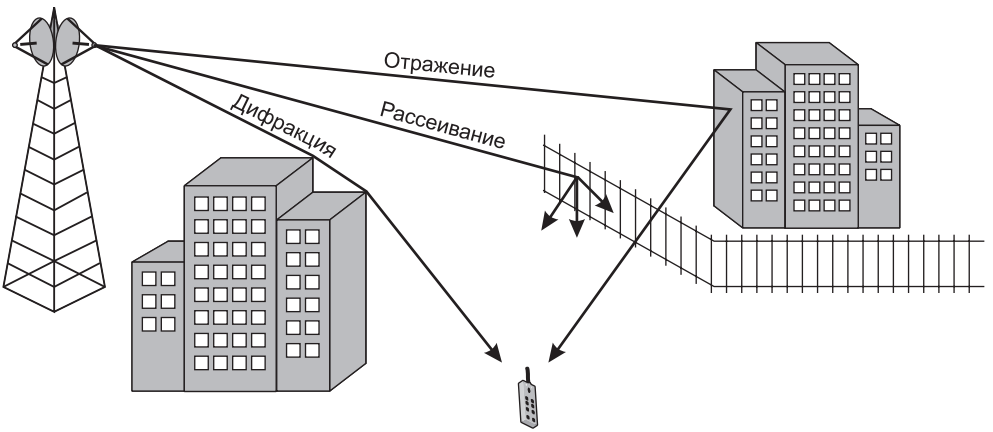


Рис. 23.3. Отражение, дифракция и рассеивание электромагнитной волны

Типы волн

Идеальной средой распространения электромагнитных волн является вакуум, однако в реальной жизни сигналы чаще передаются через атмосферу, которая является нестабильной и неоднородной средой, состоящей из многих слоев, обладающих разными проводящими свойствами. Свойства реальной передающей среды в сочетании с частотными характеристиками передаваемых сигналов определяют несколько основных способов распространения электромагнитных волн (рис. 23.4):

- земные волны;
- ионосферные волны;
- прямые волны;
- тропосферные волны.

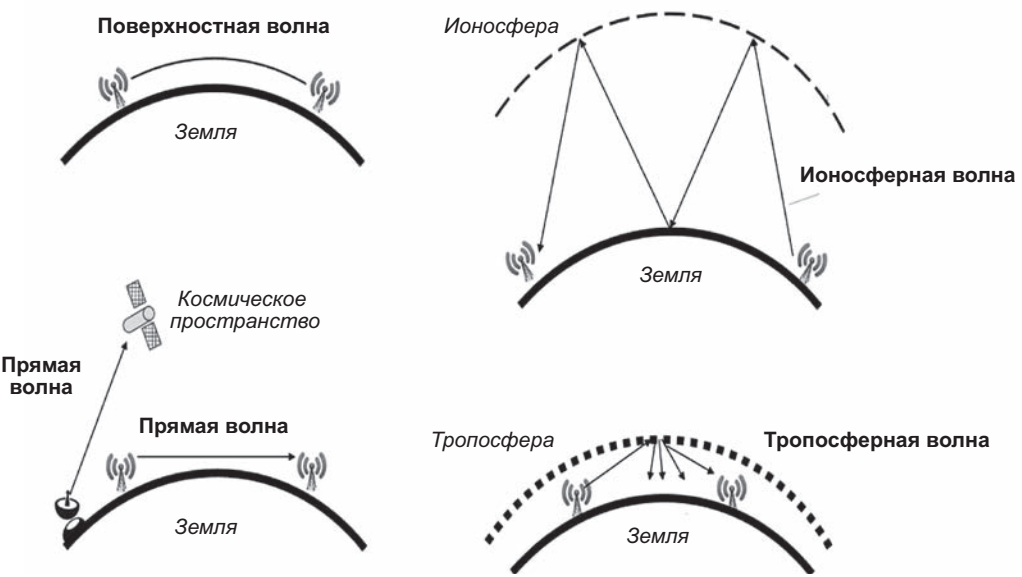


Рис. 23.4. Способы распространения электромагнитных волн

Земные, или **поверхностные**, **волны** распространяются вдоль земной поверхности. Следуя более-менее рельефу местности, они могут проходить большие расстояния, до *нескольких сотен километров*, далеко за линию видимого горизонта. Этот способ распространения волн характерен для электромагнитного излучения *низкой частоты — до 2 МГц*. Электромагнитные волны этой частоты рассеиваются в атмосфере таким образом, что не проникают в ее верхние слои. Самым известным примером земной волны является сигнал АМ-радио из диапазона длинных волн. Основной причиной того, что волны следуют поверхности земли, является *дифракция*. В данном случае непроницаемым препятствием намного большего размера, чем длина волны, является выпуклость земли. Способность волны огибать препятствие зависит от соотношения длины волны и размера препятствия; чем меньше это отношение, тем слабее проявляется дифракция. Отсюда

понятно, что для электромагнитных сигналов высокой частоты эффектом дифракции можно пренебречь.

Ионосферные, или **пространственные**, **волны** характерны для сигналов *средних и высоких частот от 2 до 30 МГц*. Сигналы, излучаемые базирующейся на земле антенной, *отражаются* ионосферой (менее плотным ионизированным верхним слоем атмосферы, занимающим уровни от 48 км до 100 км от поверхности Земли) на землю, и потому могут распространяться далеко за видимый горизонт, на расстояния, даже большие, чем поверхностные волны. При достаточной мощности передатчика радиоволны этих диапазонов за счет многократного отражения от ионосферы могут даже обогнуть земной шар. Ионосферные волны широко используются в радиовещании и в особенности международном радиовещании — например, такими компаниями, как Би-би-си (BBC Radio World Service).

Прямые волны, как это следует из их названия, распространяются только по прямой. Такой тип распространения волн свойственен электромагнитным сигналам с частотой *выше 30 МГц* — они не могут ни отражаться ионосферой, ни огибать выпуклости Земли. При частоте свыше 4 ГГц они начинают поглощаться водой, а это означает, что не только дождь, но и туман могут стать причиной резкого ухудшения качества передачи микроволновых систем. Инфракрасный и видимый свет могут быть переданы только вдоль прямой видимости, так как эти волны не проходят через стены.

Тропосферные волны относятся к тому же диапазону (30 МГц – 3 ГГц), что и прямые волны, и как было сказано, они не могут отражаться ионосферой. Однако они способны распространяться путем преломления и *рассеяния* на неоднородностях тропосферы — ближайшем к земле слое атмосферы. Тропосферные неоднородности — это области пространства, воздух в которых в некоторые моменты времени имеет температуру, давление и влажность, отличающиеся от средних для окружающей среды значений. Такие эффекты в тропосфере наблюдаются не всегда, а только при определенном состоянии атмосферы. Но если такое состояние наблюдается, то тропосферные волны позволяют передавать сигнал, хотя и весьма слабый, на *расстояние до 1000 км*.

Прямая и непрямая видимость

На распространение радиоволн существенно влияет не только их частота, но и *взаимное положение передатчика и приемника*, и наличие препятствий между ними. В зависимости от расположения связываемых объектов применяется та или иная методика конструирования системы беспроводной связи, в которой используются те или иные приемы борьбы с искажениями, например, для связи в городе используются релейные вышки, а для связи за пределами прямой видимости используется спутниковая связь. Различают следующие сценарии взаимного расположения передатчика и приемника, более точно — их передающей и принимающей антенн:

- прямая видимость (**LoS** — Line of Sight);
- непрямая видимость (**NLoS** — non Line of Sight);
- за пределами прямой видимости (**BLoS** — Beyond Line of Sight).

Прямая видимость: приемник и передатчик лежат на воображаемой прямой линии без каких-либо визуальных препятствий, и это расстояние может быть довольно большим для

невооруженного глаза, однако оно ограничено кривизной Земли. В этом случае радиоволны любой частоты и любого типа при условии достаточной мощности передатчика беспрепятственно доходят до приемника. Прямая видимость может иметь место как для наземных передатчиков и приемников, так и для радиопередающих устройств, расположенных в космическом пространстве, например, когда дрон связывается со спутником. Прямая связь на Земле ограничена расстоянием, что обусловливает низкие задержки и высокую надежность при передаче данных.

Непрямая видимость: приемник и передатчик находятся не на линии прямой видимости, между ними имеются препятствия (дома или деревья) или детали рельефа (холмы, горы). Сталкиваясь с преградами, радиоволны рассеиваются, отражаются, ослабляются. Препятствия по-разному сказываются на волнах разных частот: от пренебрежимо малых помех до полного разрыва связи. В частности, *прямые волны* (не путать с прямой видимостью) полностью блокируются препятствиями. Для борьбы с этим явлением применяются различные методы, в том числе использование одного или нескольких релейных устройств, расположенных выше преграды. На рис. 23.5 показано, как проблема непрямой видимости устраняется с помощью дронов, выполняющих функции релейных устройств.

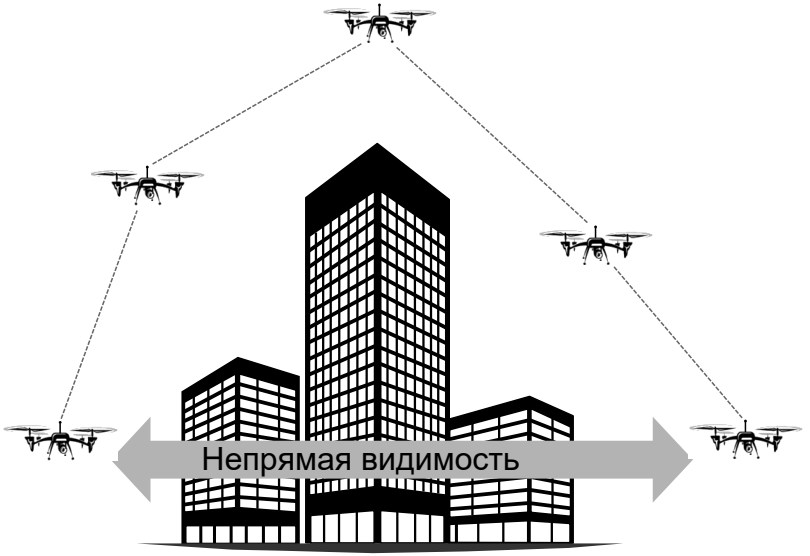


Рис. 23.5. Обеспечение связи при непрямой видимости

Приемник и передатчик находятся друг от друга **за пределами прямой видимости**, то есть за горизонтом или слишком плотно закрыты друг от друга рельефом местности. Передатчик и приемник находятся так далеко друг от друга, что препятствием становится даже кривизна Земли. Поэтому для связи абсолютно необходимы промежуточные ретрансляторы (например, спутники или самолеты). Используется также распространение радиоволн по земной поверхности и ионосферное распространение для увеличения дальности связи до нескольких тысяч километров. Например, военные дроны связываются с центрами управления, находящимися за тысячи километров, по спутниковым каналам связи.

Борьба с искажениями сигнала в беспроводных линиях связи

Отказ от проводов и обретение мобильности приводит к высокому уровню помех в беспроводных линиях связи. Если интенсивность битовых ошибок (BER) в проводных линиях связи равна 10^{-9}–10^{-10}, то в беспроводных линиях связи она достигает величины 10^{-3}! В городских условиях в диапазоне частот полезного сигнала обычно имеется большое количество помех, например от систем зажигания автомобилей, от различных бытовых приборов.

В результате дифракции, отражения и рассеяния электромагнитных волн, которые повсеместно встречаются при беспроводной связи в городе, приемник может получить несколько реплик одного и того же сигнала, прошедших к приемнику разными путями. Такой эффект называется **многолучевым распространением сигнала** (multipath signal propagation). При каждом отражении сигнал может изменять фазу, амплитуду и угол прибытия на приемник. Результат многолучевого распространения сигнала часто оказывается отрицательным, поскольку сигналы могут прийти в противофазе и подавить основной сигнал.

Так как время распространения сигнала вдоль различных путей является в общем случае различным, то может также наблюдаться **межсимвольная интерференция** — ситуация, когда в результате задержки сигналы, кодирующие соседние биты данных, доходят до приемника в течение интервала времени, отведенного для приема одного символа. Сигнал, полученный в результате наложения соседних сигналов, приемник может декодировать неверно.

Искажения из-за многолучевого распространения приводят к ослаблению сигнала — этот эффект называется **многолучевым замиранием** (fading). Известно, что при распространении электромагнитных волн в свободном пространстве (без отражений) затухание мощности сигнала пропорционально произведению квадрата расстояния от источника сигнала на квадрат частоты сигнала. В городах многолучевое замирание приводит к тому, что ослабление сигнала становится пропорциональным не квадрату расстояния, а его кубу или даже четвертой степени!

Искажения радиосигналов происходят также из-за **эффекта Доплера**, который заключается в изменении наблюдаемой частоты (или длины) волны при относительном движении источника и приемника сигнала.

Когда источник радиоволн движется к вам, каждая новая волна излучается из более близкой к вам точки, то есть волны «сжимаются». Когда же он движется от вас, волны «растягиваются». Это относится не только к электромагнитным волнам, но и к звуковым волнам, и даже волнам на воде. Например, когда пожарная машина приближается к вам, звук ее сирены начинает звучать выше, а когда она проезжает мимо и удаляется, высота звука резко понижается. Поскольку звук на выходе сирены не меняется, этот эффект изменения тона связан с изменением скорости перемещения машины относительно вас. Меняется частота, с которой звуковые волны достигают ваших ушей, а частота — это то, что наш мозг интерпретирует как высоту звука. В радиосвязи эффект Доплера приводит к тому, что приемник перестает устойчиво распознавать сигналы передатчика, так как их частота отклоняется от номинальной. Чем быстрее движутся относительно друг друга приемник и передатчик, а также чем выше частота передаваемого сигнала, тем значительнее искажение частоты. Негативные последствия эффекта Доплера особенно заметны в низко-

орбитальных спутниковых сетях, а также в мобильных сотовых сетях для пользователей, движущихся с большой скоростью в автомобилях, поездах и самолетах.

Оценку эффекта Доплера для радиоволн микроволнового диапазона можно сделать с помощью следующей формулы:

$$\Delta f = f \times v/c,$$

где f — частота сигнала на выходе передатчика, v — скорость перемещения приемника относительно передатчика, c — скорость света, а Δf — смещение частоты на входе приемника из-за эффекта Доплера.

Проблема высокого уровня помех беспроводных каналов решается различными способами. Важную роль играют рассматриваемые далее **технологии широкополосного сигнала**. Эти технологии основаны на распределении энергии сигнала в широком диапазоне частот, так что узкополосные помехи не оказывают существенного влияния на сигнал в целом. Для распознавания сигнала, искаженного из-за его многолучевого распространения, применяются различные способы обработки, компенсирующие межсимвольную интерференцию. Одним из них является **адаптивное выравнивание сигнала** (adaptive equalizing, рис. 23.6).

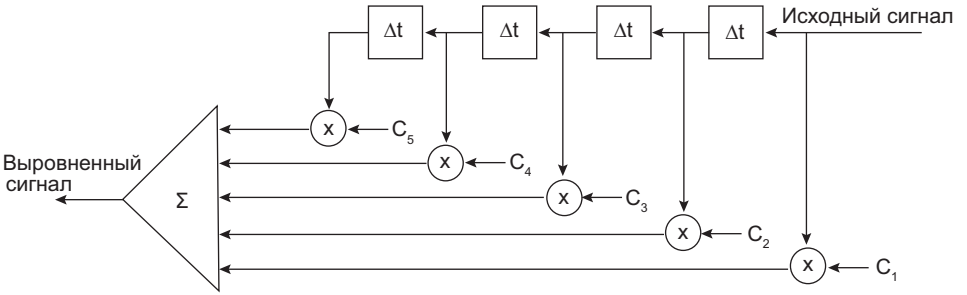


Рис. 23.6. Адаптивное выравнивание сигнала

Идея заключается в суммировании сигнала, измеренного через равные промежутки времени Δt в течение одного такта передачи символа кода. Перед суммированием значения сигнала умножаются на свой *весовой коэффициент* C_i. Значение сигнала, полученного после суммирования и называемого **выровненным сигналом**, считается значением бита переданного кода в данном такте.

Выбор веса выполняется *адаптивно*, с использованием заранее известного двоичного кода, называемого **тренировочной последовательностью**. Передатчик вставляет эту последовательность после каждого блока пользовательских данных определенной длины. Приемник применяет к тренировочной последовательности тот же алгоритм выравнивания, что и к пользовательским данным, сравнивает значение полученной последовательности бит с ожидаемой тренировочной последовательностью; если они отличаются, то вычисляются новые значения весовых коэффициентов.

Большую роль играет и применение **самокорректирующихся кодов FEC**. Радиосвязь всегда была пионером этой техники — частота возникновения битовых ошибок здесь гораздо выше, чем в проводной передаче денных. Еще одним приемом является применение **протоколов с установлением соединения** и повторными передачами кадров на *канальном*

уровне стека протоколов. Эти протоколы позволяют быстрее корректировать ошибки, так как работают с меньшими значениями тайм-аутов, чем корректирующие протоколы *транспортного* уровня, такие как TCP. И наконец, передатчики сигнала (и приемники, если это возможно) стараются разместить на высоких башнях (мачтах), чтобы избежать многократных отражений.

Лицензирование

Проблема разделения электромагнитного спектра между потребителями требует *централизованного* регулирования. В каждой стране есть специальный государственный орган, который (в соответствии с рекомендациями ITU) выдает **лицензии** операторам связи на использование определенной части спектра, достаточной для передачи информации по определенной технологии. Лицензия выдается на определенную территорию, в пределах которой оператор задействует закрепленный за ним диапазон частот монопольно.

Существуют также несколько частотных диапазонов, 433 Мгц, 860 МГц (Европа), 900 МГц (Северная Америка), 2,4 ГГц, 5 ГГц и 6 МГц, которые рекомендованы ITU как диапазоны для международного использования *без лицензирования*[1]. Эти диапазоны выделены промышленным товарам беспроводной связи общего назначения, например устройствам блокирования дверей автомобилей, научным и медицинским приборам. В соответствии с назначением эти диапазоны получили название **ISM**-диапазонов (Industrial, Scientific, Medical — промышленность, наука, медицина). Диапазоны 860 и 900 МГц являются наиболее «населенными», поскольку низкочастотная техника всегда стоила дешевле. Сегодня активно осваивается диапазон 2,4 ГГц, например, в технологиях IEEE 802.11 и Bluetooth. Обязательным условием использования этих диапазонов на совместной основе является ограничение максимальной мощности передаваемых сигналов уровнем 1 Ватт. Это условие сокращает радиус действия устройств, чтобы их сигналы не стали помехами для других пользователей, которые, возможно, задействуют тот же диапазон частот в других районах города. Кроме того, в Европе в диапазонах 433 и 860 МГц существует ограничение на так называемый «цикл работы» (duty-cycle) устройства, который определяется как усредненный в течение часа процент занятия радиоэфира передачей своей информации и равен 1 %.

Ограничения по мощности и циклу работы влияют на технологии беспроводных сетей, предназначенные работать в диапазонах частот с этими ограничениями.

Мобильные сотовые сети работают в лицензированных диапазонах частот.

Антенны

Поскольку при беспроводной связи информационные сигналы передаются не в медном проводнике или оптоволокне, а в атмосфере или любой другой *ненаправленной* среде распространения электромагнитных колебаний, то обязательным элементом беспроводной линии связи является антенна.

Антенна — устройство, предназначенное для приема и передачи электромагнитных волн. Если линия связи предназначена для работы в обоих направлениях, то одна и та же антенна

[1] Диапазоны 900 МГц и 5 ГГц свободны от лицензирования не во всех странах.

может использоваться как для приема, так и для передачи, при условии, что характеристики поступающих и излучаемых электромагнитных волн одинаковы.

Каждый узел беспроводной сети имеет беспроводной сетевой адаптер, который включает в себя приемник и/или передатчик, связанные с антенной.

При передаче источник генерирует модулированные сигналы — например, переменный электрический ток в соответствии с потоком передаваемых битов. Этот сигнал проходит по токопроводящим частям антенны, в результате чего вокруг них образуется переменное электромагнитное поле, порождающее электромагнитную волну, распространяющуюся от антенны в пространство.

При приеме происходит обратный процесс: энергия электромагнитной волны, падающей на антенну, возбуждает ток в токопроводящих элементах антенны, который поступает в приемник. Приемник выполняет декодирование полученного информационного сигнала.

Изотропные и направленные антенны

Заметим, что антенна не усиливает сигнал, но она может направлять излучение преимущественно в каком-то направлении. Как, например, человек, говорящий в простейший рупор, не может превзойти своих голосовых возможностей, но, направив все звуковые волны в желаемом направлении, он может сделать свое выступление более громким. Такой же эффект достигается при использовании абажуров, концентрирующих освещение в тех или иных местах комнаты.

Антенна, излучающая волны одинаково во всех направлениях, называется **всенаправленной**, или **ненаправленной**, или **изотропной** антенной. Однако чаще антенны излучают сигналы разной интенсивности в разных направлениях. Они называются **направленными** антеннами. Заметим, что распространение излучения во всех направлениях можно также обеспечить несколькими направленными антеннами.

Хотя, как отмечено, антенны не усиливают сигнал, существует характеристика антенны, называемая коэффициентом усиления. В данном случае **коэффициент усиления антенны**, или **коэффициент направленного действия антенны** (antenna gain), означает отношение выходной мощности, излучаемой данной антенной в определенном направлении, к выходной мощности, излучаемой идеальной изотропной антенной в любом направлении, при условии, что на вход обеих антенн поступают с передатчиков равные по мощности сигналы.

Коэффициент усиления антенны К измеряется в децибелах, то есть $K = 10 \lg (p/P_{\text{изотр.}})$, где p — мощность выходного сигнала тестируемой антенны в некоторой точке пространства, а $P_{\text{изотр.}}$ — мощность изотропной антенны. Например, если коэффициент усиления некоторой антенны равен 3 дБ, то $\lg(p/P_{\text{изотр}}) = 0{,}3$, $p/P_{\text{изотр}} = 10^{0{,}3}$, что составляет примерно 2. Другими словами, если коэффициент усиления интересующей нас антенны в некотором направлении равен 3 дБ, то излучаемая ею мощность в этом направлении в два раза превосходит мощность излучения эталонной изотропной антенны. Такой выигрыш в мощности получен не за счет усиления, а за счет *перераспределения* энергии сигнала путем уменьшения излучения в других направлениях. Графическое представление зависимости коэффициента усиления антенны от пространственных координат называют **диаграммой направленности** антенны. Диаграммы являются трехмерными объектами, но часто более удобно работать с их двумерными проекциями. Единицы, в которых измеряется коэффи-

циент усиления антенны, иногда помечают индексом «и» или «i» (дБи или dBi), показывающим, что это отношение к мощности изотропной (isotropic) антенны.

Так как излучение *идеальной* изотропной антенны одинаково по всем направлениям, ее диаграмма направленности является сферой, в центре которой находится излучатель (рис. 23.7, *а*). На рисунке показана проекция сферы, из центра которой к поверхности направлены два произвольных вектора, имеющих одинаковую длину, что показывает, что мощности сигнала в соответствующих точках равны: $P_1 = P_2$. Заметим, что коэффициент направленного действия идеальной изотропной антенны равен 0.

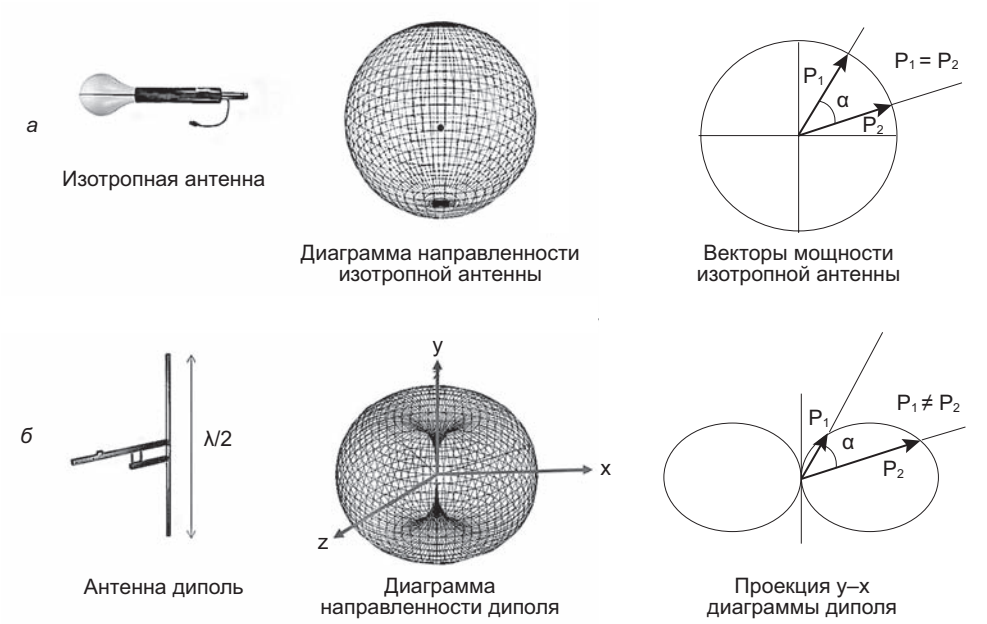


Рис. 23.7. Диаграммы направленности изотропной антенны и антенны диполя

На рис. 23.7, *б* показана антенна **диполь** (антенна Герца), состоящая из двух равных соосных отрезков проводника — проводов или стержней, разделенных небольшим промежутком. Диаграмма направленности диполя имеет форму тора и в целом весьма близка к сферической диаграмме изотропной антенны. Действительно, если смотреть на антенну вдоль оси, по которой расположены два горизонтальных отрезка антенны, то в поперечном сечении (плоскость z—x) диаграммы направленности мы увидим такой же круг (он не показан на рисунке), как и для идеальной изотропной антенны. То есть в направлениях, перпендикулярных оси антенны, сигнал излучается равномерно во все стороны. В вертикальном разрезе (плоскость y—x) диаграмма имеет форму восьмерки. Здесь в направлении, указываемом вектором P_1, излучается меньшая мощность, чем в направлении P_2. То есть в этой плоскости диполь излучает неравномерно: чем ближе к оси антенны, тем меньше излучение. Обычно диполь относят к слабонаправленным или всенаправленным антеннам.

Антенной с ярко выраженной направленностью является **параболическая антенна**. Источник излучения (рис. 23.8) помещается в фокусе параболического отражателя. В теории

лучи, исходящие из фокуса параболы, отражаются поверхностью антенны параллельно друг другу. Чем больше диаметр антенны, тем более направленным является излучаемый сигнал. Антенны такого типа широко используются в качестве домашних спутниковых антенн («тарелок»), в радиотелескопах и в спутниковых системах связи.

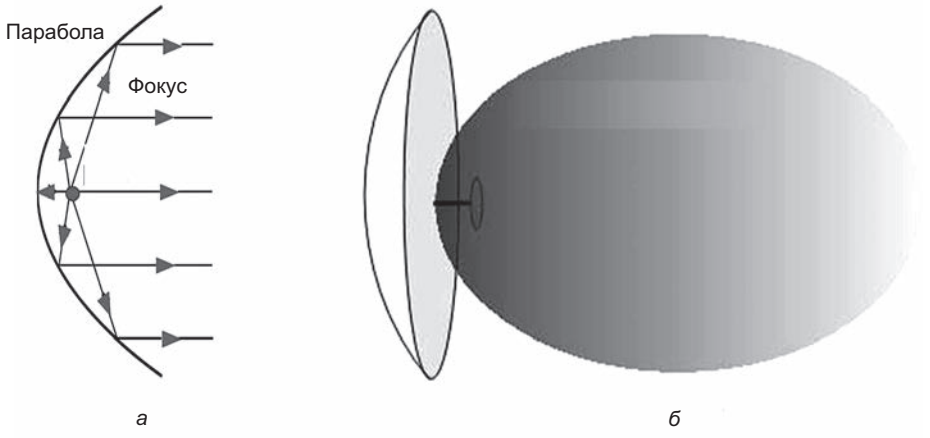


Рис. 23.8. *а* — отражательные свойства параболической поверхности; *б* — диаграмма направленности параболической антенны

Имея диаграмму направленности антенны, можно легко определить преимущественное направление, в котором эта антенна излучает электромагнитные волны. Угол, в пределах которого мощность, излучаемая антенной, не меньше чем половина мощности, излучаемой в наиболее преимущественном направлении, называют **шириной диаграммы направленности** или **углом излучения антенны** (beam width). Для принимающей антенны угол излучения указывает на направление, способствующее наилучшему приему.

Процесс излучения антенны зависит от длины ее излучающих частей или, точнее, от того, как эта длина соотносится с длиной волны излучаемого или принимаемого электромагнитного сигнала. Например, диполь будет наиболее эффективно принимать/передавать сигналы с длиной волны λ, если он имеет длину $\lambda/2$. Другим примером является антенна монополь (антенна Маркони) — в простейшем случае это вертикально устанавливаемый отрезок провода или стержня длиной $\lambda/4$, где λ — длина волны принимаемого сигнала (такого типа антенны использовались, например, в портативных радиоприемниках). Поскольку при ненаправленном распространении электромагнитные волны заполняют все пространство (в пределах определенного радиуса, определяемого затуханием сигнала), это пространство может служить *разделяемой средой*[1]. Разделение среды передачи порождает те же проблемы, что и в локальных сетях, однако здесь они усугубляются тем, что пространство, в отличие от кабеля, является общедоступным, а не принадлежит одной организации.

[1] О разделяемой среде передачи см. главу 2.

Прием и передача с использованием нескольких антенн (MIMO)

Для повышения скорости, дальности передачи и помехозащищенности линий связи используются разнообразные методы, включая повышение мощности сигнала, различные способы модуляции и кодирования, частотное и временное мультиплексирование, наложение фильтров и др. Наряду со всеми этими возможностями еще одним действенным ресурсом для улучшения характеристик беспроводного обмена данными стало *увеличение числа антенн* (и соответственно передатчиков и приемников) при построении линии связи.

Беспроводные технологии, разработанные для эффективного использования в одной линии связи нескольких передатчиков и приемников, называют технологиями **MIMO** (Multiple-In Multiple-Out, буквально «несколько входов, несколько выходов»). Термин MIMO часто используется в более узком смысле, о чем будет сказано ниже. Технологии MIMO существуют уже не один десяток лет, но сейчас, с появлением мобильных сетей, особенно поколений 4G и 5G, они переживают второе рождение. MIMO используются и в локальных сетях: уже стало обычным наличие 3–4 антенн в точках доступа Wi-Fi (начиная со стандарта Wi-Fi 4).

Конфигурации систем с несколькими антеннами

В зависимости от числа антенн линии связи могут иметь следующие конфигурации (рис. 23.9):

- **MIMO** (Multiple-In Multiple-Out) — наиболее общая конфигурация линии связи, имеющая несколько передающих и несколько приемных антенн, остальные конфигурации представляют собой частные случаи MIMO; обозначение NxM MIMO означает систему с N передатчиками и M приемниками.
- **SISO** (Single-Input Single-Output) — традиционная схема беспроводной линии с одним передатчиком и одним приемником.
- **MISO** (Multiple-Input Single-Output) — линия связи с несколькими передатчиками и одним приемником. Основная задача в системах такой конфигурации — формирование оптимального выходного сигнала путем цифровой настройки передающих антенн.
- **SIMO** (Single-Input Multiple-Output) — линия связи с одним передатчиком и несколькими приемниками. Сигнал с одной антенны передатчика поступает в приемник в виде нескольких реплик, полученных в результате многолучевого распространения. Главной задачей для такой конфигурации является извлечение приемником дополнительной информации из сигналов, поступивших на избыточные антенны, и ее использование для более качественного воспроизведения исходного сигнала.

В широком смысле термин MIMO относится ко всем конфигурациям, кроме SISO, а в узком — только к одной из них, собственно MIMO, которая единственная обладает структурой, имеющей потенциал для повышения пропускной способности линии связи за счет организации передачи *несколькими параллельными потоками данных*. Избыточные антенны систем MISO и SIMO не могут быть использованы для параллельной передачи, а значит, и для повышения пропускной способности линии, однако они могут служить для улучшения качества передачи данных.

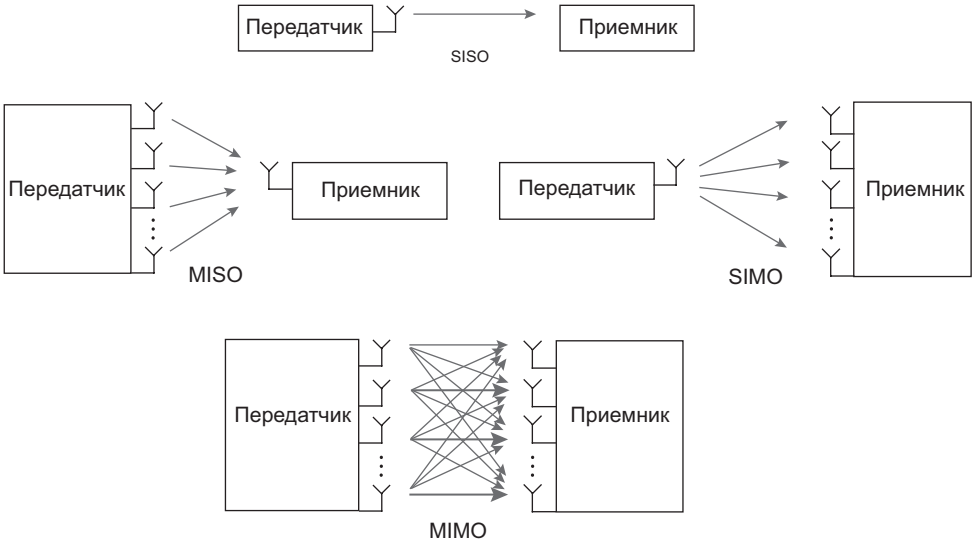


Рис. 23.9. Варианты конфигурации систем в зависимости от числа антенн

Еще одна конфигурация — многопользовательский **MU-MIMO** (Multi-User MIMO)— соответствует ситуации, когда на приемной стороне работает несколько пользователей, каждый из которых в общем случае оснащен несколькими антеннами (рис. 23.10). В большинстве беспроводных линий связи типа «один ко многим» (например, связь точки доступа сети Wi-Fi с мобильными устройствами пользователей[1]) в каждый момент времени обслуживается запрос только одного пользователя. Системы MU-MIMO частично снимают это ограничение, поддерживая несколько одновременных информационных потоков с разными пользователями. Параллельные потоки могут быть организованы и в обратном направлении — от мобильного устройства одного пользователя к нескольким передающим

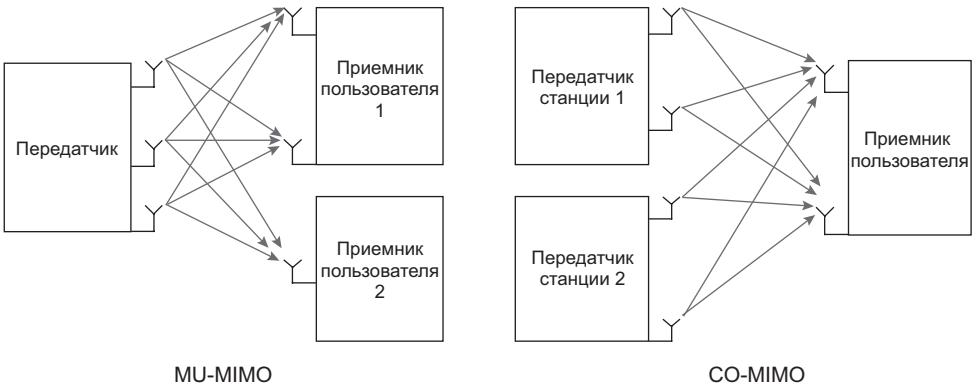


Рис . 23.10. Конфигурации систем MIMO с несколькими пользователями и несколькими передающими станциями

[1] См. главу 22.

станциям (Cooperative MIMO, или **CO-MIMO**). Запрос на такого типа связи особенно высок в современных мобильных сетях.

Технологии MIMO можно разделить на два класса (рис. 23.11):

- Технологии, базирующиеся на концепции **пространственного разнесения**. В этих технологиях несколько антенн принимают или передают разные реплики одних и тех же информационных сигналов; увеличение числа антенн служит повышению качества приема, однако скорость передачи ненамного выше, чем в системах SISO.
- Технологии, базирующиеся на концепции **пространственного мультиплексирования**. Они предназначены для организации в одном канале передачи нескольких независимых информационных потоков, за счет чего повышается пропускная способность линии связи, но возрастает вероятность ошибки.



Рис. 23.11. *а*) пространственное разнесение; *б*) пространственное мультиплексирование

Пространственное разнесение

В основе идеи использования нескольких антенн при пространственном разнесении (Spatial Diversity, SD) лежит эффект **многолучевого распространения сигнала** — в отсутствие прямой видимости передаваемый сигнал доходит до приемника в виде нескольких реплик, каждая из которых пришла в приемник *своим путем*, отражаясь от различных препятствий (стен, потолков, поверхности земли, слоев атмосферы и др.). В результате отражений в каждую реплику исходного сигнала вносятся временны́е задержки, фазовые сдвиги, затухание и другие искажения. В зависимости от сочетания искажений в разных репликах, при их наложении в приемнике могут возникать разные эффекты — как конструктивные, так и деструктивные. Например, синусоидальные сигналы, сдвинутые по фазе на величину, близкую π, взаимно компенсируют друг друга, а на 2π — вдвое усиливают друг друга. Ранее многолучевое рассеяние рассматривалось только в негативном аспекте, как источник помех и причина замедления передачи. Но, оказывается, *многолучевое распространение можно использовать конструктивно*, что и сделано в технологиях MIMO.

Рассмотрим простой пример. Если на приемнике имеется только одна антенна, то в отсутствие информации о путях прохождения сигнала сложно предсказать заранее, как скажется на принятом сигнале многолучевой эффект — не исключено, например, что это будет замирание. Если же установить на приемнике несколько антенн, то можно гарантировать, что возникновение замирания одновременно на всех них принципиально исключается, а значит, можно выбрать один сигнал, лучший по определенному критерию, например, имеющий наибольшую мощность. В более развитых системах, например, использующих метод оптимального весового сложения (MaximalRatio Combining, MRC), приемник выполняет для полученных сигналов процедуру адаптивного выравнивания, затем сравнивает их, назначая каждому весовой коэффициент в соответствии с его качеством. Результирующий сигнал получается путем взвешенного сложения и обладает улучшенным соотношением сигнал/шум, что в некоторых случаях позволяет увеличить скорость передачи. Таким

образом, наличие нескольких реплик сигнала, принятых несколькими антеннами, в совокупности дает значительный объем дополнительной информации, которая теоретически делает возможной организацию более надежного канала связи, чем в системах SISO. Проиллюстрируем сказанное простой аналогией. Каждую антенну на приемной стороне можно сравнить со свидетелем некоего происшествия. Субъективные показания каждого отдельного свидетеля содержат много искажений, но следователь (приемник), учитывая и сопоставляя все свидетельства (реплики сигналов с каждой антенны), строит более адекватную картину произошедшего (исходный сигнал).

Пространственное разнесение как прием, заключающийся в использовании нескольких антенн для повышения качества и надежности беспроводной линии связи, может использоваться как на приемной стороне — разнесенный прием (как в примере выше), так и на стороне передающих систем — разнесенная передача (в последнем случае множество антенн формируют оптимальный выходной сигнал). Чем больше пространственно разнесенных реплик сигнала имеется в распоряжении приемника, тем более качественную картину он может из них извлечь. Если беспроводной канал имеет N передающих антенн и M приемных антенн, то максимально возможное количество «разнесений» равно числу каналов передачи, то есть произведению $N \times M$.

Примером пространственного разнесения на передающей стороне является адаптивная передача. Адаптивность заключается в том, что приемник формирует на каждой из выходных антенн сигналы с такими характеристиками, которые, поступив на антенны приемника, обеспечивают максимально возможное высокое качество приема. Для решения этой задачи передатчик выполняет предварительную процедуру калибровки каналов передачи, включающую запрос к приемнику о том, с каким качеством к нему поступают сигналы с каждой из передающих антенн. Совокупность характеристик путей составляет **информацию о состоянии канала** (Channel State Information, **CSI**). CSI характеризует различные свойства канала: затухание мощности, распределение замирания, коэффициент усиления антенны и др. На основании полученной от приемника информации передатчик вычисляет **матрицу параметров**, элементы которой соответствуют всем сочетаниям (номер передающей антенны, номер принимающей антенны), то есть если передатчик имеет N антенн, а приемник — M антенн, то матрица состояния каналов имеет размерность $N \times M$. Параметры могут, в частности, включать амплитуду, фазу, мощность для каждого сигнала. Затем передатчик использует эту матрицу для предварительной модификации сигналов для каждой антенны для улучшения качества результирующего сигнала, получаемого от нескольких передающих антенн на приемной стороне. Такая обработка сигналов перед отправкой их на антенны передатчика называется **прекодированием**. В отличие от метода оптимального весового сложения, в котором основная тяжесть принятия решений и обработки сигналов возлагается на приемную сторону, при адаптивной передаче активную роль играет передающая сторона. Информация CSI периодически передается от приемников сигнала к передатчикам сигнала, образуя *обратную связь,* которая и создает основу адаптивного поведения систем MIMO.

Рассмотренные выше технологии MIMO, использующие несколько антенн для пространственного разнесения, в основном нацелены на повышение качества передачи, то есть увеличение соотношения сигнал/шум — $P_с/P_ш$. Рост числа антенн дает лишь *медленное* увеличение пропускной способности за счет увеличения отношения сигнал-шум в соответствии с формулой Шеннона: $C/F \sim \log_2(1 + N\, P_с/P_ш)$, где N — число антенн.

(S) *Пространственно-временно́е кодирование STC*

Пространственное мультиплексирование

При **пространственном мультиплексировании** (Spatial Multiplexing, **SM**) наличие нескольких антенн у передающего и принимающего узла используется для создания нескольких параллельных независимых потоков данных. Очевидно, что такое применение системы MIMO существенно повышает скорость передачи данных. Распараллеливание потока данных всегда являлось одним из наиболее эффективных способов повышения пропускной способности линий связи. Мы видели это на примере многих технологий проводных сетей — Ethernet, SDH, OTN, DWDM.

Увеличить пропускную способность линии связи можно и другими способами, например, увеличив мощность сигнала или уменьшив уровень помех. Но влияние этих параметров ограничено логарифмической зависимостью. Так, для увеличения спектральной эффективности метода передачи только на 1 бит/с (то есть увеличения битовой скорости на 1 бит/с в расчете на 1 Гц полосы частот) требуется увеличить мощность сигнала примерно вдвое. К тому же во многих областях применения беспроводной связи существуют как технические, так и законодательные ограничения на мощность передающих устройств. Расширение полосы частот также не всегда возможно из-за административных ограничений.

В рассмотренных ранее способах мультиплексирования, основанных на разделении времени (TDM), частотного диапазона (FDM) или даже волны (WDM), каждому потоку данных выделяются либо временные слоты, либо полоса частот. При пространственном мультиплексировании все потоки передаются в той же самой полосе частот и квантование времени отсутствует. В данном случае разделяемым ресурсом является *пространство* — исходный поток данных разделяется на несколько независимых подпотоков, каждый из которых передается отдельной антенной в своем «пространственном слоте».

Пропускная способность системы MIMO, работающей по технологии SM, растет *линейно* с ростом числа антенн $C/F \sim K \log_2(1 + P_c/P_ш)$, где $K = \min\{N, M\}$ — максимальное количество пространственных потоков, N — число антенн передатчика, M — число антенн приемника. Если после образования пространственных потоков на приемнике остаются избыточные антенны, то они могут быть использованы для улучшения качества сигнала методами пространственного разнесения.

При использовании пространственного мультиплексирования SM беспроводные системы MIMO могут работать в двух режимах — однопользовательском (просто MIMO) и многопользовательском MU-MIMO. В режиме MU-MIMO выходные потоки передающего узла направляются на антенны принимающих узлов, принадлежащих разным пользователям и, в общем случае, находящихся в разных точках пространства. Обычно такой режим характерен для центральных узлов беспроводных сетей, таких как базовые станции мобильных сотовых сетей или точек доступа сетей Wi-Fi, обслуживающих населенные пользователями области (аэропорты, стадионы и т. п.). Такие системы называют **массивными MIMO** (Massive MIMO), для них характерно использование большого количества антенн (не менее 32). В случае, когда устройство пользователя также поддерживает технику MIMO, обычно с гораздо меньшим числом антенн, с двумя или четырьмя, то массивная система MIMO распараллеливает потоки данных как между пользователями, так и между антеннами приемника такого пользователя.

Хотя теоретически увеличение числа антенн в *n* раз должно приводить к увеличению пропускной способности системы MIMO SM в *n* раз, в реальности рост оказывается

более медленным из-за того, что пространственные каналы не являются абсолютно независимыми и вносят взаимные помехи. В ряде случаев, когда уровень шумов и замираний в канале весьма высок, метод пространственного мультиплексирования может вообще не дать ожидаемого выигрыша в пропускной способности из-за необходимости частых повторных передач ошибочных данных.

Чтобы этого не случилось, в системах MIMO и MU-MIMO применяются методы формирования диаграммы направленности отдельных антенн массива, а также предварительное кодирование сигналов, посылаемых на отдельные антенны.

Формирование диаграммы направленности и предварительное кодирование

Для формирования **диаграммы направленности**, называемого также **формированием луча** (Beamforming), в системах MIMO и MU-MIMO применяются **фазированные антенные решетки**. Такая антенна представляет собой набор небольших индивидуальных антенн, количество которых может измеряться десятками, а иногда и сотнями. За счет изменения амплитуды и фазы сигнала каждой индивидуальной антенны для каждого приемника генерируются пространственно направленные сигналы. Массив динамических антенн создает диаграммы направленности как в горизонтальном, так и в вертикальном направлении, что позволяет фокусировать энергию в нескольких отдельных точках пространства, в которых находятся конечные узлы пользователей MU-MIMO-системы (рис. 23.12).

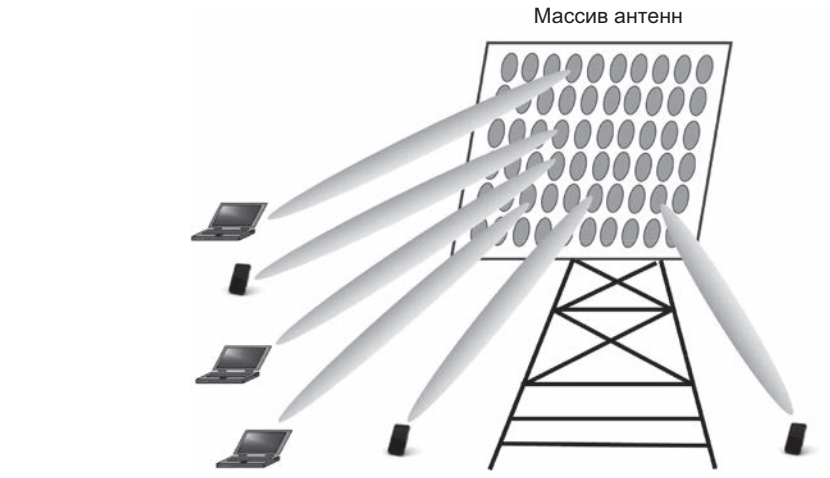


Рис. 23.12. Формирование диаграммы направленности системой Massive MIMO

Фокусировка луча выполняется путем выбора оптимальных значений фазы составляющих сигналов на основании информации о состоянии канала передач CSI.

Формирование диаграммы направленности может выполняться как *аналоговыми*, так и *цифровыми* средствами. При аналоговом формировании радиолуча один и тот же сигнал подается на каждую антенну решетки, а затем аналоговые фазовращатели управляют сигналом, излучаемым всем массивом антенн. При цифровом формировании луча на разные

антенны решетки подаются разные сигналы, так достигается бóльшая гибкость — разным антеннам можно назначать не только разные фазы и коэффициенты усиления, но также частотные полосы.

Информация о состоянии канала CSI также используется для пре-кодирования сигналов антенн MIMO и MU-MIMO. Сигналы кодируются таким образом, чтобы символы кода для различных антенн были ортогональными, то есть каждая приемная антенна массива MIMO устойчиво распознавала символы своего потока данных, а символы других потоков считала шумом. Детали техники пре-кодирования мы здесь рассматривать не будем из-за их сложности. Ограничимся только замечанием, что при одновременном использовании методов формирования луча и пре-кодирования приемник не занимается работой по демультиплексированию потоков, так как только сигналы его потока, попадающие на его антенну, имеют достаточную мощность, а сигналы остальных потоков оказываются слишком слабыми. Поэтому задача приемника ограничивается декодированием этих сигналов, то есть превращением принятых символов сигнала в двоичный код.

Нужно подчеркнуть, что как при формировании луча, так и при пре-кодировании сигналов, в формировании сигнала для каждой принимающей антенны принимают участие *все передающие антенны*. Поэтому изображенное на рис. 23.12 взаимодействие массива антенн MU-MIMO с устройствами пользователей нужно считать условным, отражающим цель техники MU-MIMO, а не ее реализацию.

Бюджет беспроводной линии связи

Напомним, что бюджетом линии связи называется расчет баланса всех усилений и потерь мощности сигнала вдоль пути сигнала по линии связи от передатчика до приемника. Расчет бюджета линии позволяет оценить уровень мощности сигнала на входе приемника и проверить, является ли он достаточным для надежного распознавания передаваемой по линии связи информации.

Техника расчета бюджета беспроводной линии связи остается такой же, как и для проводной линии, которую мы рассмотрели в главе 6, но составляющие бюджета отличаются (рис. 23.13):

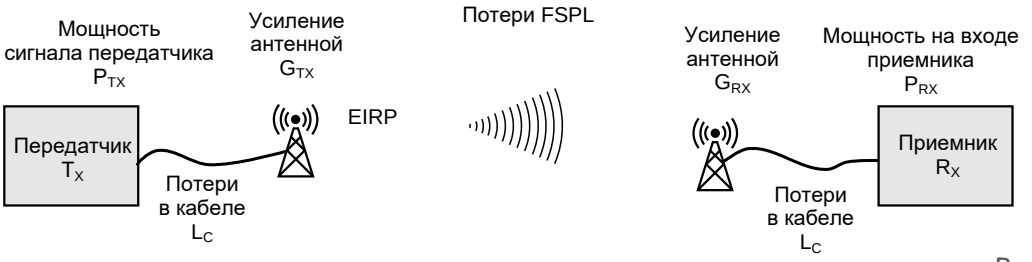


Рис. 23.13. Составляющие бюджета беспроводной линии

- P_{TX} — мощность сигнала, генерируемого передатчиком (дБм);
- L_c — потери в кабеле, соединяющем передатчик с антенной (дБ);
- G_{TX} — усиление антенной передатчика (дБи);

- EIRP (Effective Isotropic Radiated Power) — мощность сигнала, генерируемого антенной передатчика в определенном направлении;
- FSPL (Free Space Path Loss) — потери в свободном пространстве (дБ);
- G_{RX} — усиление антенной приемника (дБи);
- P_{RX} — мощность сигнала на входе приемника (дБм).

Специфика беспроводной линии связи проявляется в двух составляющих бюджета — усиление антенны и потери в свободном пространстве. Первую мы уже рассмотрели, теперь остановимся на второй.

Потери мощности при распространении в свободном пространстве (то есть когда сигнал не искажается ни по какой из названных ранее причин) происходят из-за распределения мощности сигнала по все более увеличивающейся поверхности фронта сигнала по мере удаления от антенны. Сферическая модель распространения сигнала в свободном пространстве дает следующую формулу для вычисления его затухания:

$$FSPL = 10 \lg (4\pi d/\lambda)^2,$$

где λ — длина волны колебаний сигнала, d — расстояние от источника сигнала (антенны).

Для упрощения расчетов эту формулу обычно представляют в другом виде:

$$FSPL = 32{,}44 + 20 \lg (d) + 20 \log (f),$$

где $f = 1/\lambda$, частота колебаний сигнала в мГц, d измеряется в километрах, а значение постоянной составляющей зависит о выбранных единиц измерения, если использовать, например, кГц и метры, то оно будет другим.

Из этой формулы видно, что затухание увеличивается пропорционально квадрату частоты сигнала, поэтому низкочастотные несущие проникают дальше и проектировщик беспроводной сети всегда стоит перед выбором: использовать ли высокочастотные сигналы, чтобы увеличить скорость передачи данных, или же предпочесть низкочастотные для увеличения расстояния между узлами.

Параметр EIRP в расчете бюджета линии является промежуточным:

$$EIRP = P_{TX} - L_c + G_{TX},$$

он показывает мощность сигнала, излучаемого антенной, иногда он известен для данной аппаратуры (передатчик плюс антенна). Именно этот параметр имеют в виду регуляторы телекоммуникационных услуг, когда ограничивают мощность передатчика телефона или другого устройства пользователя.

В принятых нами обозначениях расчет бюджета линии выглядит так:

$$P_{RX} = P_{TX} - Lc + G_{TX} - FSPL + G_{RX} - L_c.$$

Проверка достаточности мощности сигнала на входе состоит в том, чтобы сравнить полученную мощность сигнала на входе приемника с его порогом чувствительности.

Порог чувствительности приемника обычно указывается его производителем и зависит от многих факторов, одним из которых является величина термального шума на входе приемника. **Термальный шум** — это электронный шум, генерируемый случайным движением электронов внутри электрического проводника. Об этом факторе полезно помнить при вы-

боре ширины полосы пропускания для передаваемого модулированного сигнала, так как термальный шум прямо пропорционален этой ширине, так что приемнику труднее распознавать широкополосные сигналы из-за более высокого уровня термального шума. В электронном оборудовании всегда присутствует термальный шум, и его величина ограничивает порог чувствительности приемника, который не может быть меньше уровня термального шума.

Пример расчета бюджета линии

Пусть мы имеем: P_{TX} = 20 дБм (100 мВ), L_c = 1 дБ, G_{TX} = 8 дБ, d = 1 км, G_{RX} = 2 дБ, f = 2400 МГц (типичная частота Wi-Fi-сети).

Для расстояния 1 км нужно подсчитать потери в свободном пространстве:

$$FSPL = 32, 44 + 20 \lg(1) + 20 \lg (2400) = 100,04.$$

Тогда мощность сигнала на входе приемника будет равна:

$$P_{RX} = 20 - 1 + 8 - 100, 04 + 2 - 1 = -72,04 \approx -72 \text{ (дБ)}.$$

Пусть чувствительность приемника равна –82 дБ, тогда наш расчет показывает, что бюджет линии достаточен для устойчивого распознавания сигналов приемником, так как у него есть запас в 10 дБ для устойчивой работы.

Техника расширенного спектра

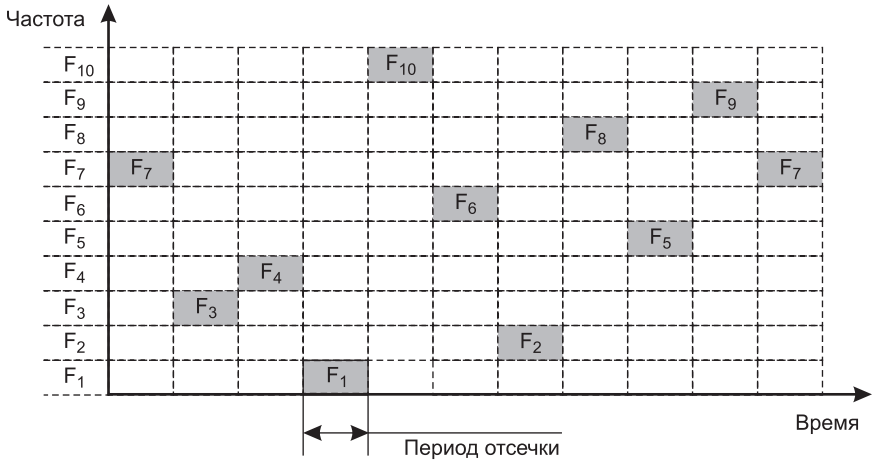
Техника расширенного спектра разработана специально для беспроводной передачи и позволяет повысить помехоустойчивость кода для сигналов малой мощности за счет увеличения *спектра передаваемого сигнала*. Сигналы расширенного спектра более устойчивы к узкополосным внешним помехам, возникающим в беспроводной среде в случайные моменты времени и на различных случайных частотах. Подчеркнем: эти искажения имеют другую природу, нежели искажения, возникающие на краях диапазона из-за обрезания боковых гармоник сигнала слишком узкой полосой пропускания передающей среды и ограничивающие пропускную способность канала. Расширение спектра полезного сигнала приводит к тому, что внешняя помеха искажает только несколько его гармоник, но сохранившиеся неискаженными гармоники позволяют правильно распознать сигнал. Вместе с тем расширение спектра сигнала отдельного пользователя приводит к нежелательному эффекту — необходимости выделения для каждого пользователя более широкой полосы частот радиоэфира. Чтобы компенсировать этот нежелательный эффект, методы расширения спектра дополняются специфическими *методами мультиплексирования сигналов* отдельных пользователей, которые и позволяют достичь баланса между двумя целями — обеспечением помехоустойчивости передачи и наиболее эффективным разделением общего диапазона частот между многочисленными пользователями беспроводной сети.

Далее рассматриваются несколько основных методов расширения спектра.

Расширение спектра скачкообразной перестройкой частоты FHSS

Идея метода **расширения спектра скачкообразной перестройкой частоты** (Frequency Hopping Spread Spectrum, **FHSS**) возникла во время Второй мировой войны, когда радио

широко использовалось для секретных переговоров и управления военными объектами, например торпедами. Чтобы радиообмен нельзя было перехватить или подавить узкополосным шумом, было предложено вести передачу с постоянной сменой несущей в пределах широкого диапазона частот. В результате мощность сигнала распределялась по всему диапазону и прослушивание какой-то определенной частоты давало только небольшой шум. Последовательность несущих частот выбиралась псевдослучайной, известной только передатчику и приемнику. Попытка подавления сигнала в узком диапазоне также не слишком ухудшала сигнал, поскольку подавлялась только небольшая часть информации (рис. 23.14).



Последовательность перестройки частот: F_7-F_3-F_4-F_1-F_{10}-F_6-F_2-F_8-F_5-F_9

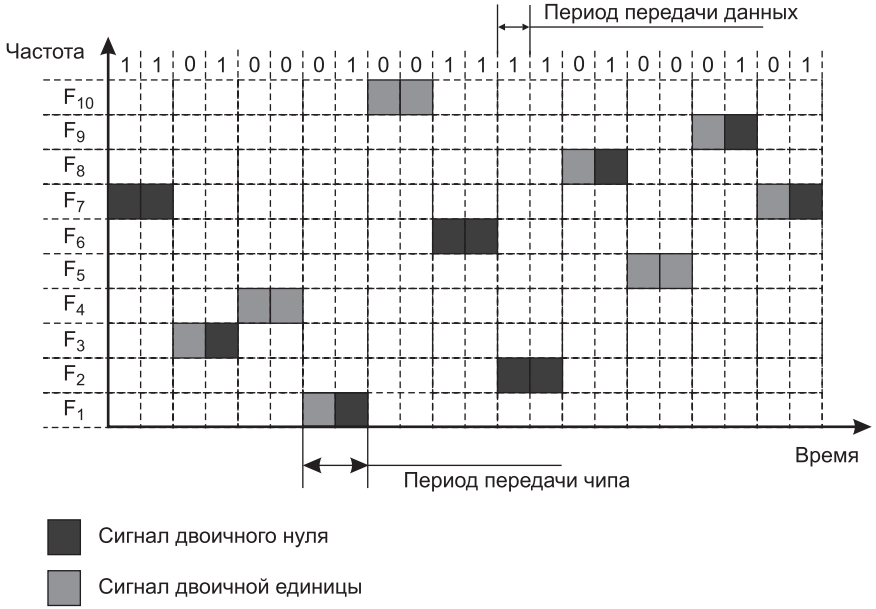
Рис. 23.14. Расширение спектра скачкообразной перестройкой частоты

Несущая частота меняется в соответствии с номерами частотных подканалов, вырабатываемых алгоритмом псевдослучайных чисел. Псевдослучайная последовательность зависит от некоторого параметра, который называют **начальным числом**. Если приемнику и передатчику известны алгоритм и значение начального числа, то они меняют частоты в одинаковой последовательности.

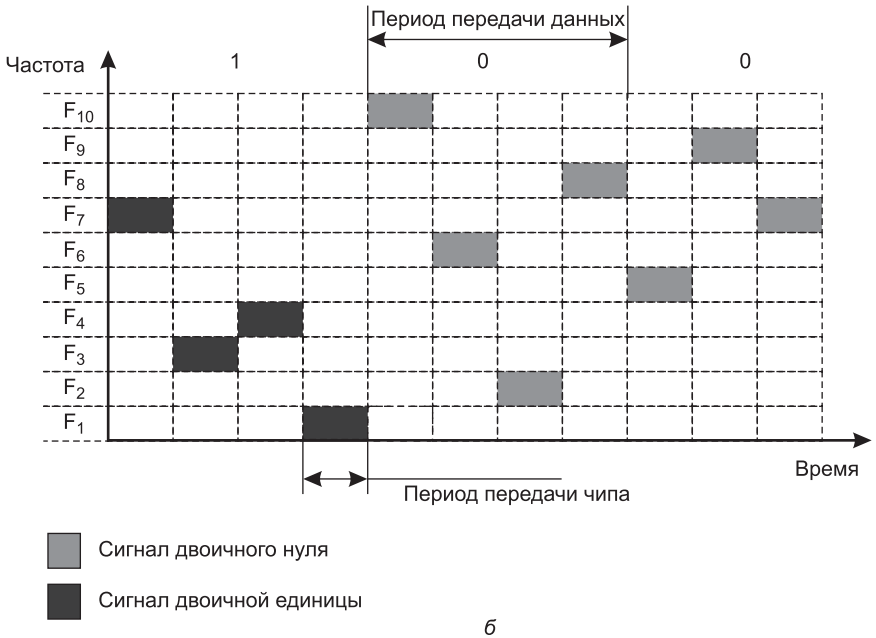
На каждой несущей частоте для передачи дискретной информации применяются стандартные методы модуляции: FSK или PSK. Чтобы приемник синхронизировался с передатчиком, для обозначения начала каждого периода передачи некоторое время передаются синхробиты. Таким образом, полезная скорость этого метода кодирования оказывается меньше из-за постоянных накладных расходов на синхронизацию. Если частота смены подканалов ниже, чем скорость передачи данных в канале, то такой режим называют **медленным расширением спектра** (рис. 23.15, *а*); в противном случае мы имеем дело с **быстрым расширением спектра** (рис. 23.15, *б*).

Метод быстрого расширения спектра более устойчив к помехам, поскольку узкополосная помеха, подавляющая сигнал в определенном подканале, не приводит к потере бита, так как его значение передается несколько раз в различных частотных подканалах. В этом режиме не проявляется эффект межсимвольной интерференции, потому что ко времени

прихода задержанного вдоль одного из путей сигнала система успевает перейти на другую частоту. Метод медленного расширения спектра таким свойством не обладает, но зато он проще в реализации и менее дорог.



а



б

Рис. 21.15. Соотношение между скоростью передачи данных и частотой смены подканалов

Методы FHSS применяют в беспроводных технологиях IEEE 802.11 (Wi-Fi) и Bluetooth. Отметим, в методах FHSS подход к использованию частотного диапазона не такой, как в других методах кодирования, — вместо экономного расходования узкой полосы делается попытка занять весь доступный диапазон. На первый взгляд это кажется не очень эффективным. Однако если учесть, что сигнал отдельного пользователя в течение периода времени между сменой частоты использует не весь диапазон, а узкий спектральный подканал, то напрашивается и более эффективное решение по совместному использованию доступного диапазона *несколькими пользователями*. Это решение состоит в том, что частотные подканалы каждого пользователя изменяются в псевдослучайном порядке, при этом в каждый момент времени пользователи передают свои сигналы в различных частотных подканалах. Конечно, это можно сделать, только если число пользователей не превышает числа частотных подканалов. Таким образом, выполняется мультиплексирование пользователей при использовании метода FHSS.

Прямое последовательное расширение спектра DSSS

В методе **прямого последовательного расширения спектра** (Direct Sequence Spread Spectrum, **DSSS**) частотный диапазон расширяется не за счет постоянных переключений с частоты на частоту, как в методе FHSS, а за счет того, что каждый бит информации заменяется N битами, поэтому тактовая скорость передачи сигналов увеличивается в N раз. Это, в свою очередь, означает, что спектр сигнала также расширяется в N раз. Достаточно соответствующим образом выбрать скорость передачи данных и значение N, чтобы спектр сигнала заполнил весь диапазон.

> Код, которым заменяется двоичная единица исходной информации, называется **расширяющей последовательностью**. Двоичный нуль кодируется *инверсным* значением расширяющей последовательности. Приемники должны знать расширяющую последовательность, которую использует передатчик, чтобы понять передаваемую информацию.

Примером расширяющей последовательности является *последовательность Баркера* (Barker), которая состоит из 11 бит: 10110111000. Если передатчик использует эту последовательность, то передача трех бит 110 ведет к отправке следующих битов:

10110111000 10110111000 01001000111.

Последовательность Баркера позволяет приемнику быстро синхронизироваться с передатчиком, то есть надежно выявлять начало последовательности. Приемник определяет такое событие, поочередно сравнивая получаемые биты с образцом последовательности. Действительно, если сравнить последовательность Баркера с такой же последовательностью, но сдвинутой на один бит влево или вправо, то мы получим меньше половины совпадений значений битов. Значит, даже при искажении нескольких битов, с большой долей вероятности приемник правильно определит начало последовательности, а значит, сможет правильно интерпретировать получаемую информацию.

Биты кода расширения (в примере 10110111000) принято называть **чипами** (от англ. chips — кусочки) для того, чтобы их можно было отличать от битов исходных пользовательских данных (в примере 110). Скорость передачи чипов в радиосреде, называемая

чиповой скоростью, выше, чем битовая скорость данных. Чиповая скорость пропорциональна количеству битов в расширяющей последовательности — эта пропорция называется **коэффициентом расширения**. Чем больше коэффициент расширения, тем шире спектр результирующего сигнала и тем больше степень подавления помех. Обычно коэффициент расширения имеет значение от 10 до 100. Как и в случае FHSS, для кодирования битов результирующего кода может использоваться любой вид модуляции, например BFSK.

Множественный доступ с кодовым разделением CDMA

В мобильных сетях широко используется техника **множественного доступа с кодовым разделением** (Code Division Multiplexing Access, **CDMA**). В предыдущем разделе, рассматривая разделение канала в системах MIMO для организации параллельной передачи нескольких потоков, мы добавили к ранее изученным способам новый способ — пространственное мультиплексирование. Расширим этот список методов мультиплексирования еще одним — CDMA, в котором информационным потокам выделяются не кванты времени, не частотные каналы и не пространственные слоты. В этом методе им назначаются *различающиеся коды*.

> Идея CDMA заключается в том, что каждый узел сети задействует *собственное* значение расширяющей последовательности, выбираемое так, чтобы принимающий узел, который знает значение расширяющей последовательности передающего узла, мог выделить данные передающего узла из суммарного сигнала, образующегося в результате одновременной передачи информации несколькими узлами.

Каждый узел сети, работающий по методу CDMA, посылает данные в разделяемую среду в те моменты времени, когда это ему нужно, то есть *синхронизация между узлами отсутствует*. В CDMA каждый бит данных исходного потока, поступающий в разделяемую среду, заменяется на приписанную данному узлу расширяющуюся последовательность. Коды расширения не являются произвольными битовыми последовательностями — они должны обладать особым свойством *ортогональности*, под которым в данном случае понимается возможность выделения кода отдельного узла из общего сигнала. Для построения кодов расширения привлекаются математические методы. Полученный в результате замены код изменяется с более высокой чиповой скоростью, чем исходная последовательность.

Поясним идею CDMA на примере. Пусть в сети работает четыре узла: *A*, *B*, *C* и *D*. Каждый узел использует следующие значения расширяющей последовательности:

A: 0 1 0 1 0 1 0 1
B: 1 0 1 0 0 1 0 1
C: 1 0 0 1 1 0 0 1
D: 1 1 1 1 1 1 1 1

Предположим также, что при передаче единиц и нулей расширяющей последовательности (то есть уже преобразованного исходного кода) используются сигналы, которые являются аддитивными и инверсными. Инверсность означает, что двоичная единица кодируется, например, синусоидой с амплитудой $+A$, а двоичный ноль — синусоидой с амплитудой $-A$. Из условия аддитивности следует, что если фазы этих амплитуд совпадут, то при одно-

временной передаче единицы и нуля мы получим нулевой уровень сигнала. Для упрощения записи расширяющей последовательности обозначим синусоиду с положительной амплитудой значением +1, а синусоиду с отрицательной амплитудой — значением –1. Для простоты допустим также, что все узлы сети CDMA синхронизированы.

Таким образом, при передаче единицы исходного кода 4 узла передают в среду такие последовательности:

A: –1 +1 –1 +1–1 +1 –1 +1
B: +1 –1 +1 –1 –1 +1 –1 +1
C: +1 –1 –1 +1 +1 –1 –1 +1
D: +1 +1 +1 +1 +1 +1 +1 +1

При передаче нуля исходного кода сигналы расширяющей последовательности инвертируются.

Пусть теперь каждый из четырех узлов независимо от других передает в сеть один бит исходной информации: узел $A \rightarrow 1$, узел $B \rightarrow 0$, узел $C \rightarrow 0$, узел $D \rightarrow 1$. В среде *S* сети наблюдается такая последовательность сигналов:

A: –1 +1 –1 +1–1 +1 –1 +1
B: –1 +1 –1 +1 +1 –1 +1 –1
C: –1 +1 +1 –1 –1 +1 +1 –1
D: +1 +1 +1 +1 +1 +1 +1 +1

В соответствии со свойством аддитивности получаем:

S: –2, +4, 0, +2, 0, +2, +2, 0

Если, например, некоторый узел *E* хочет принимать информацию от узла *A*, то он должен использовать свой демодулятор CDMA, задав ему в качестве параметра значение расширяющей последовательности узла *A*. Демодулятор CDMA последовательно складывает все четыре суммарных сигнала S_i, принятые в течение каждого такта работы. При этом сигнал S_i, принятый в такте, на котором код расширения станции *A* равен +1, учитывается в сумме со своим знаком, а сигнал, принятый в такте, на котором код расширения станции *A* равен –1, добавляется в сумму с противоположным знаком. Другими словами, демодулятор выполняет операцию скалярного умножения вектора принятых сигналов на вектор значения расширяющей последовательности нужной станции:

$$S \times A = (-2, +4, 0, +2, 0, +2, +2, 0) \times (-1\ +1\ -1\ +1-1\ +1\ -1\ +1) = 8.$$

Чтобы узнать, какой бит послала станция *A*, нормализуем результат, то есть разделим его на количество разрядов в расширяющей последовательности: 8/8 = 1.

Если бы станция хотела принимать информацию от станции *B*, то ей нужно было бы при демодуляции использовать код расширения станции *B* (+1 –1 +1 –1 –1 +1 –1 +1):

$$S \times B = (-2, +4, 0, +2, 0, +2, +2, 0) \times (+1\ -1\ +1\ -1\ -1\ +1\ -1\ +1) = -8.$$

После нормализации получаем сигнал –1, который соответствует двоичному нулю исходной информации станции *B*.

Мы объяснили только основную идею CDMA, предельно упростив ситуацию. На практике CDMA — весьма сложная технология, оперирующая не условными значениями +1 и –1, а модулированными сигналами, например сигналами BPSK. Проблема синхронизации приемника и передатчика решается за счет передачи длинной последовательности определенного кода — **пилотного сигнала**.

Ортогональное частотное мультиплексирование

Еще одним способом расширения спектра сигнала является **ортогональное частотное мультиплексирование** (Orthogonal Frequency Division Multiplexing, **OFDM**). Этот метод похож на обычное частотное мультиплексирование FDM (см. главу 7), поскольку и здесь применяется мультиплексирование сигналов в различных частотных полосах, называемых **каналами**.

> Однако вместо того чтобы передавать данные отдельной станции в одном частотном канале с максимально возможной скоростью, в OFDM эти данные передаются небольшими частями *параллельно и независимо* по нескольким частотным полосам, называемых *подканалами*, с более низкой скоростью. В отличие от FDM, в котором канал разделяется между несколькими пользователями, все подканалы одного канала OFDM передают данные *одного* потока.

И хотя скорость передачи в отдельных подканалах может быть невысокой, за счет одновременной передачи данных по всем подканалам достигается высокая результирующая общая скорость. Идея распараллеливания скоростного потока на несколько низкоскоростных дополняется в технологии OFDM более ***плотным***, чем в технологии FDM, размещением подканалов в диапазоне частот (рис. 23.16).

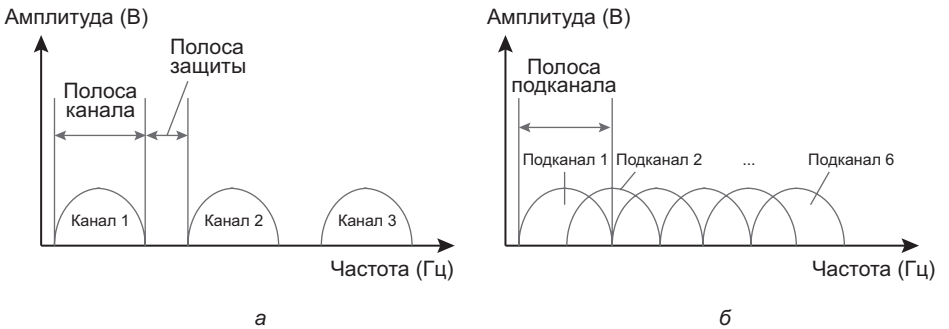


Рис. 23.16. Взаимное расположение каналов FDM (*а*) и подканалов OFDM (*б*)

На рисунке показано взаимное расположение каналов технологии FDM (*а*) и подканалов технологии OFDM (*б*). Каждому каналу технологии FDM отводится полоса частот определенной ширины, позволяющая передавать модулированные сигналы определенного спектра, умещающегося в эту полосу. Чтобы сигналы соседних каналов не искажали друг друга за счет взаимодействия боковых гармоник, каналы разделены ***полосой защиты***. Как видно из рисунка, в технологии OFDM подканалы не только не разделены полосой защиты, а, напротив, сдвинуты навстречу друг другу на половину полосы подканала, и в одном

и том же диапазоне частот помещается более чем в 2 раза больше подканалов OFDM, чем каналов FDM.

Возможность качественной передачи данных при таком плотном расположении частотных каналов обеспечивается тем, что разбиение канала передачи на подканалы и обработка передаваемых в пределах этих подканалов сигналов выполняется *специальным* образом, обеспечивающим их *ортогональность*.

В OFDM взаимно ортогональными должны быть функции, описывающие сигналы каждого из подканалов в частотной области. Сильно упрощая, можно сказать, что если функции некоторого вида являются ортогональными, то их произведение равно нулю. Применительно к методу мультиплексирования OFMD ортогональность означает, что перекрывающиеся сигналы подканалов взаимно аннулируются, поэтому ***полоса защиты не нужна.*** Ортогональность подканалов является необходимым условием того, чтобы из общего сигнала всегда было возможно выделить сигналы каждого подканала, и в отношении подканалов OFDM достигается за счет применения двух приемов:

- Несущие частоты подканалов должны быть смещены друг относительно друга в точности на величину частоты модуляции сигнала f_m в каждом из подканалов.
- К модулированному сигналу каждого подканала перед передачей применяется фильтр импульсной формы (pulse shaping filter), за счет чего спектр сигнала приобретает специальную форму, обеспечивающую взаимное аннулирование сигналов соседних подканалов.

На рис. 23.17, *а* показана амплитудно-частотная характеристика фильтра импульсной формы. Эта характеристика описывается формулой вида $\sin(\pi\omega)/\pi\omega$ и получила свое название по форме амплитудно-частотной характеристики отдельного прямоугольного импульса. Такой фильтр работает на определенной ***несущей*** частоте f_c и имеет ширину $2f_m$, где f_m — частота ***модуляции сигнала***. Такая ширина фильтра достаточна для передачи сигналов со скоростью $C = 2f_m$, если один символ имеет два различимых состояния (в соответствии с теоремой Найквиста $C = 2f_m \log_2 M$).

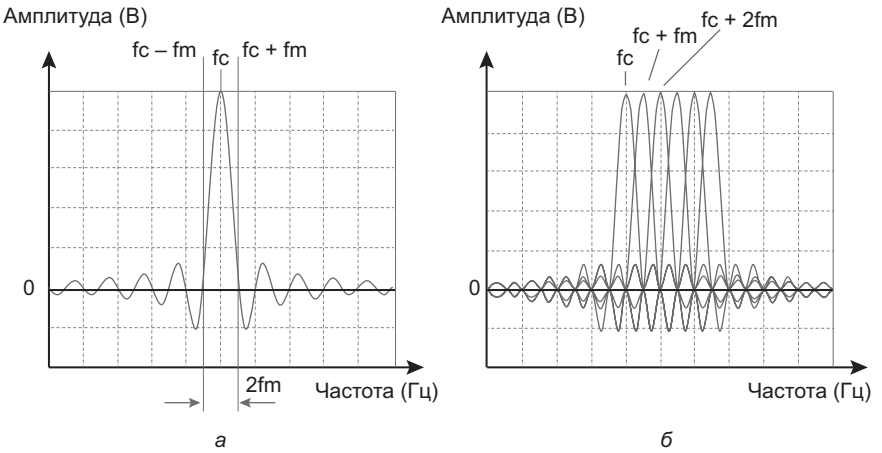


Рис. 23.17. Амплитудно-частотная характеристика фильтра импульсной формы (*а*) и амплитудночастотная характеристика подканалов OFDM (*б*)

При организации подканалов частота несущей каждого подканала отличается от частот несущей соседних подканалов на величину *частоты модуляции сигнала в подканале* f_m. Как видно из рис. 23.17, максимум амплитудно-частотной характеристики подканала совпадает с точками перехода амплитудно-частотных характеристик соседних каналов через 0, поэтому сигналы соседних каналов мало влияют на основные гармоники данного подканала, то есть подканалы являются ортогональными. Техника модуляции сигнала на каждом подканале может быть своя: например, в одном подканале может использоваться техника модуляции QPSK, а в другом — QAM16. Главное, чтобы частота модуляции у всех подканалов была *одинаковая и равная сдвигу несущей частоты*.

На рис. 23.18 показана схема, иллюстрирующая процесс преобразования кодов и сигналов в системе передатчик-приемник технологии OFDM. Несколько пояснений к рисунку.

1. На вход передатчика поступает последовательность нулей и единиц передаваемого кода. *Блок S/P* (Serial/Parallel) преобразует этот последовательный поток в несколько параллельных (в нашем примере — потоки *a*, *b*, *c* и *d*), выбирая из входного потока по нескольку битов, в зависимости от количества состояний сигнала метода модуляции, выбранного для канала. На рис. 23.18 блок S/P выбирает из входного потока по 2 бита, то есть выбран метод модуляции с четырьмя состояниями сигнала, например QPSK, а за один такт модуляции передается 2 бита исходного кода. Такт определяет частоту модуляции f_m. Блок S/P производит четыре параллельных потока данных, то есть передача исходной информации будет осуществляться по четырем подканалам системы OFDM.
2. Каждый из четырех потоков поступает на *блок модуляции Mod*, где модулируется в соответствии с выбранный техникой, например, квадратурной фазо-частотной модуляции QPSK на несущей частоте f_c конкретного подканала. Структуру квадратурного модулятора мы рассматривали в главе 7, в технологии OFDM такой модулятор обычно работает в режиме прямого цифрового синтеза DDS. Результат модуляции представлен на выходе модулятора в виде набора гармоник модулированного и отфильтрованного цифровым импульсным фильтром сигнала. Например, если в некотором такте по подканалу *a* поступает код 10, то результатом работы модулятора будет *несколько дискретных частот*, например, три числа, если после фильтрации спектр сигнала состоит из трех основных гармоник. Модулятор может иметь библиотеку спектров модулированных и отфильтрованных сигналов для различных значений входных дискретных данных, например, для кода 00 имеется один набор значений гармоник сигнала, соответствующего этому коду, для кода 01 — другой набор и т. д. Результаты преобразований условно показаны на рис. 23.18 в виде четырех значений амплитуды основной гармоники спектра каждого подканала на выходе блока Mod (спектр каждого подканала упрощен до основной гармоники, хотя в реальности он состоит из нескольких гармоник).
3. Значения дискретного спектра каждого сигнала вместе со значениями частот каждого из четырех подканалов OFDM поступают на блок IFFT (Inverse Fast Fourie Transformation), выполняющий *обратное преобразование Фурье*. Выходными сигналами блока IFFT являются функции времени, соответствующие модулированному сигналу каждого частотного подканала OFDM. Таким образом, обратное преобразование Фурье выполняет функции смесителя, перемножающего модулирующие сигналы с сигналами несущих каждого подканала.
4. Последним блоком передатчика является блок *цифро-аналогового преобразования* ЦАП. Этот блок преобразует дискретные данные, являющиеся выходными данными блока IFFT, в аналоговый сигнал, необходимый для передачи по радиоэфиру.

5. Приемник выполняет обратные преобразования сигнала, включая их преобразование из аналоговой в цифровую форму (блок АЦП), прямое преобразование Фурье (блок FFT) и их демодуляцию (блок Demod).

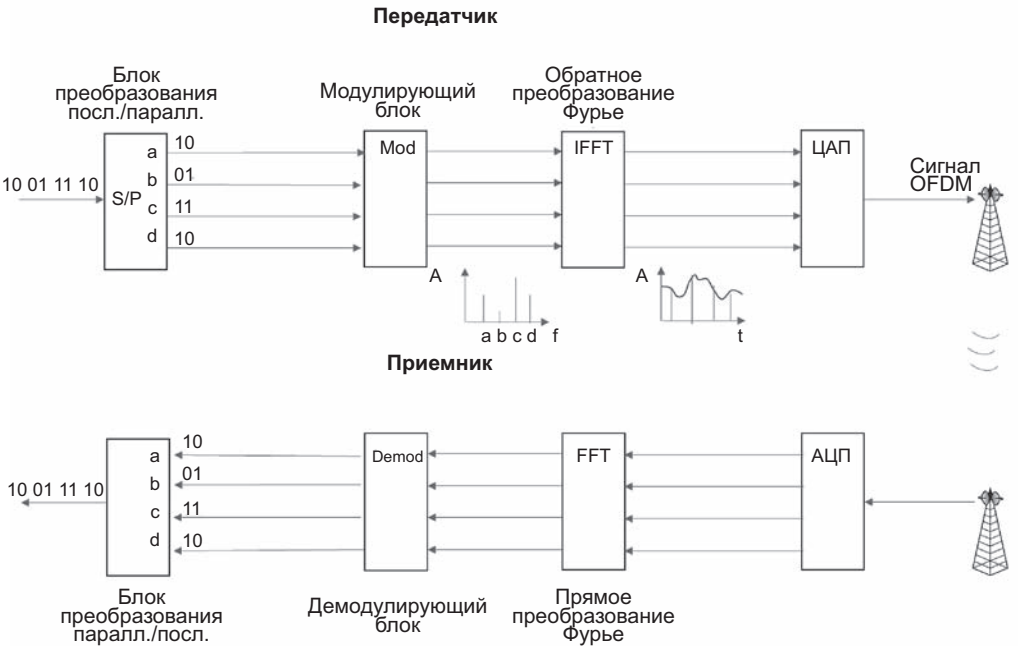


Рис. 23.18. Преобразование сигналов в системе передатчик-приемник OFDM

Передача символов кода в узких подканалах OFDM с меньшей частотой модуляции по сравнению с использованием полной полосы пропускания канала в методе FDM снижает эффект межсимвольной интерференции, происходящей из-за многолучевого распространения сигнала, так как сигнал имеет большую длительность. Для дальнейшего уменьшения межсимвольной интерференции в технологии OFDM применяется также метод, основанный на увеличении длительности передачи символа кода за счет добавления к нему **циклического префикса (Cyclic Prefix, CP)**.

(S) *Циклический префикс в OFDM*

Многопользовательский режим OFDMA

Мультиплексирование OFDM предполагает выделение всей полосы пропускания частотного канала одному пользователю, и мультиплексирование происходит между различными подпотоками данных этого пользователя. Этот режим мультиплексирования иллюстрируется рис. 23.19, на нем частотный канал сети разбит на восемь частотных подканалов и общий поток данных от пользователя соответственно распараллеливается на восемь подпотоков. Каждый подпоток передается в своем подканале, а на выходе сети подпотоки собираются в общий поток и передаются пользователю-получателю.

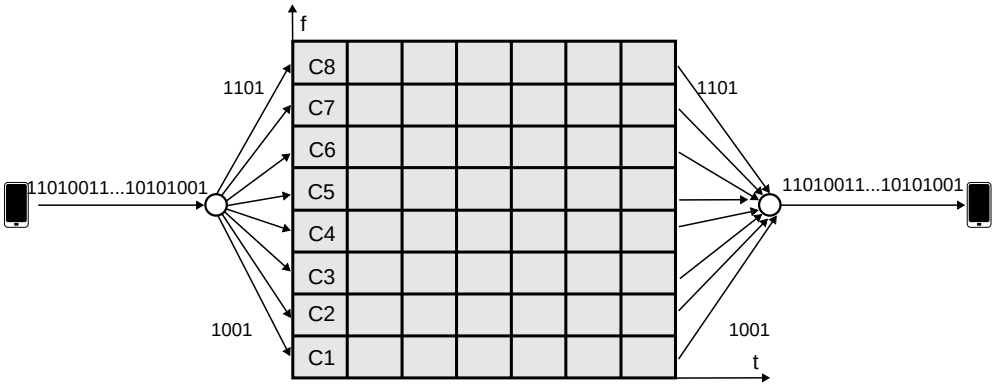


Рис. 23.19. Распределение подканалов OFDM между подпотоками одного пользователя

При применении техники OFDM мультиплексирование данных различных пользователей (в тех сетях, которые работают в многопользовательском режиме) может осуществляться на основе техники TDM за счет выделения каждому пользователю определенного тайм-слота.

Разработчики многопользовательского режима OFDMA (OFDM Access) пошли по другому пути: они решили разделять между различными пользователями не тайм-слоты, а подканалы OFDM.

Идея мультиплексирования на основе частотных подканалов иллюстрируется рис. 23.20. В соответствии с ней на вход системы OFDMA параллельно поступают несколько потоков данных от разных пользователей, в данном случае — от трех.

В нашем примере система OFDMA состоит из восьми частотных подканалов *C*1-*C*8. В начальный момент времени подканалы *C*1, *C*2 и *C*3 выделены для передачи данных первого пользователя U1, подканалы *C*4, *C*5 и *C*6 — для передачи данных пользователя *U*2, а подканалы *C*7 и *C*8 — для передачи данных пользователя *U*3.

Для передачи потока данных отдельного пользователя по нескольким выделенным ему подканалам применяется техника OFDM. Это означает, что поток распараллеливается на несколько подпотоков и каждый из них передается по отдельному подканалу. Например, в нашем примере поток данных пользователя *U*1 распараллелен на три подпотока, которые передаются в подканалах *C*1, *C*2 и *C*3.

Таким образом, в OFDMA сочетаются две техники — техника частотного мультиплексирования FDM между потоками различных пользователей и техника OFDM при передаче потока отдельного пользователя параллельными потоками по нескольким частотным подканалам.

Распределение подканалов между пользователями происходит динамически. Подканалы выделяются потоку пользователя на какое-то определенное время, например, для передачи определенного количества символов. Назовем это время периодом планирования. После завершения передачи такой порции данных распределение подканалов происходит заново, в зависимости от потребностей в передаче данных каждого пользователя (способы, каким образом эта потребность оценивается, мы сейчас не рассматриваем, потому что они могут быть разными).

На рис. 23.20 во втором периоде планирование распределения подканалов между пользовательскими потоками не изменилось, а вот при третьем планировании подканал *C*6 был отобран у потока пользователя *U*3 и отдан потоку пользователя *U*2. Вы можете сами проследить последующие решения планирования на периодах 4 и 7.

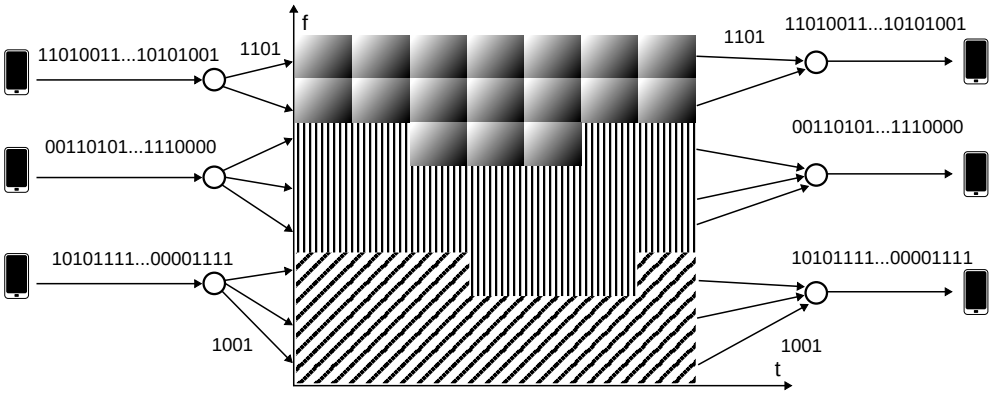


Рис. 23.20. Распределение подканалов OFDMA между тремя пользователями

Гибкое динамическое планирование распределения частотных подканалов между потоками данных пользователя является основной особенностью и основным достоинством техники мультиплексирования OFDMA. Функцию планирования обычно выполняет центральный инфраструктурный узел сети — точка доступа или базовая станция, а пользовательские устройства выполняют его решения.

Техника мультиплексирования OFDMA сегодня широко используется в беспроводных сетях разного типа, таких как Wi-Fi и мобильные сотовые сети 4G/5G.

ГЛАВА 24 Сотовые и спутниковые сети

Мобильная сотовая сеть представляет собой глобальную телекоммуникационную сеть, предоставляющую широкий спектр услуг, включающий доступ к интернету, телефонные и мультимедийные соединения и объединение в сеть многочисленных разнообразных устройств интернета вещей. Эти услуги предоставляются мобильным пользователям и мобильным устройствам.

Эволюция мобильных сотовых сетей

Сотовые телекоммуникационные сети прошли несколько этапов в своем развитии, называемых поколениями (generation), — с первого (1G) по пятое (5G).

Идея сот родилась не сразу — первые мобильные телефоны работали по другому принципу, обращаясь к одной базовой станции, покрывающей большую территорию. Примером такой системы является телефонная мобильная система «Алтай», созданная в СССР в 60-е годы, которая обслуживалась высокими мачтами и мощными радиопередатчиками. В начале 1963 года антенны центральной станции «Алтай-1» были смонтированы на шпиле высотки на Котельнической набережной[1], оборудование было размещено в помещениях здания[2].

Идея небольших сот была впервые сформулирована еще в 1945 году, но прошло довольно много времени, прежде чем заработали первые коммерческие сотовые телефонные сети — пробные участки появились в конце 60-х, а их широкое коммерческое применение началось в начале 80-х.

Первое поколение предоставляло только услуги телефонии, но, в отличие от традиционных телефонных сетей, абонент мог свободно перемещаться с мобильным телефоном по большой территории, которая являлась территорией покрытия провайдера этого абонента, не прерывая разговора при своих перемещениях, даже если он передвигался с большой скоростью, находясь в автомобиле или поезде. Так как этот тип сетей основан на использовании большого количества сот (областей покрытия), то их стали называть мобильными сотовыми сетями.

Второе поколение представлено мобильными сетями **GSM** (Global System for Mobile Communications — глобальная система для мобильных коммуникаций), которые получили большое распространение во многих странах мира. Сети GSM характеризуются несколькими революционными нововведениями:

- Передача голоса в цифровой форме (но все еще в технике коммутации каналов).
- Отдельной от голоса сетью передачи данных GPRS (General Packet Radio Service). Скорости передачи данных средствами GPRS были невелики: от 10–20 Кбит/с до 200–300 Кбит.

---

[1] https://ru.wikipedia.org/wiki/Жилой_дом_на_Котельнической_набережной

[2] https://ru.wikipedia.org/wiki/Алтай_(система_связи)

- Введение SIM-карт с данными пользователя, строгая аутентификация и шифрация трафика.

Третье поколение не было столь революционным, как второе. Основное новшество — переход на кодовое мультиплексирование CDMA. Этот метод мультиплексирования существенно повысил эффективность использования спектра частот, то есть позволил обслуживать больше пользователей в соте (при той же ширине полосы частот). Скорость передачи компьютерных данных повысилась до нескольких мегабит в секунду.

Четвертое и пятое поколения ознаменовались *полным переходом на протоколы стека TCP/IP*. Такой переход понадобился из-за изменения приоритетов пользователей, для которых доступ к услугам интернета стал не менее важным, чем телефонный разговор. В результате все типы данных современной мобильной сотовой сети передаются в IP-пакетах. Мобильные сети 5G стали не только высокоскоростными сетями доступа к интернету, но и вполне самостоятельными компьютерными сетями.

В этой главе мы изучим базовые технологии последних двух поколений сотовых сетей — 4G и 5G.

Мобильные сотовые сети 4G/LTE

Сотовые сети 4G имеют официальное название — **сети LTE (Long Term Evolution)**, из которого видно, что технология этих сетей была задумана как технология будущего, способная развиваться эволюционно, постепенно улучшая свою функциональность. Действительно, многое из того, что составляет основу технологии LTE, перешло в технологию сетей пятого поколения, а самим сетям LTE уже исполнилось 16 лет, и сегодня они являются мировым лидером по базе абонентов: в середине 2025 года их насчитывалось 4,8 миллиарда, или 53,8 % от всех абонентов сотовых сетей.

Сеть LTE разделяет все предоставляемые пользователю услуги на два класса:

- **услуги сети передачи данных** (например, доступ к интернету);
- **услуги мультимедийных соединений**, сюда входят телефонные звонки и сеансы видеосвязи.

Передача компьютерных данных и голосовых данных происходит во многом похожим образом — IP-пакеты, в которых эти два типа данных переносятся, передаются одним и тем же способом через сеть LTE. Разница состоит только в том, что в случае мультимедийных услуг сеть выполняет дополнительные действия — она находит вызываемого абонента через интернет и устанавливает с ним соединение, после этого голосовые данные передаются так же, как и компьютерные. Поэтому большая часть рассматриваемого в этой главе материала относится в равной степени к обоим типам услуг, а особенности реализации мультимедийных услуг изучаются в разделе «Передача голоса через сеть LTE».

Пользовательские характеристики сетей LTE — скорость передачи данных и задержки пакетов — вполне удовлетворяют многие приложения пользователей, обеспечивая загрузку данных из сети со скоростью до 2 Гбит/с.

В беспроводных сетях доступа 4G и 5G применяется метод мультиплексирования OFDMA, который мы рассмотрели в главе 23. Этот метод обеспечивает большее число пользователей в соте, чем другие методы мультиплексирования, применявшиеся в предыдущих поколениях сотовых сетей, а также низкие задержки и высокую спектральную плотность сигналов.

Элементы и структура сети LTE

Сотовая сеть провайдера мобильной связи является гибридной IP-сетью (рис. 24.1) и состоит из:

- беспроводной сети доступа и
- проводной магистральной сети.

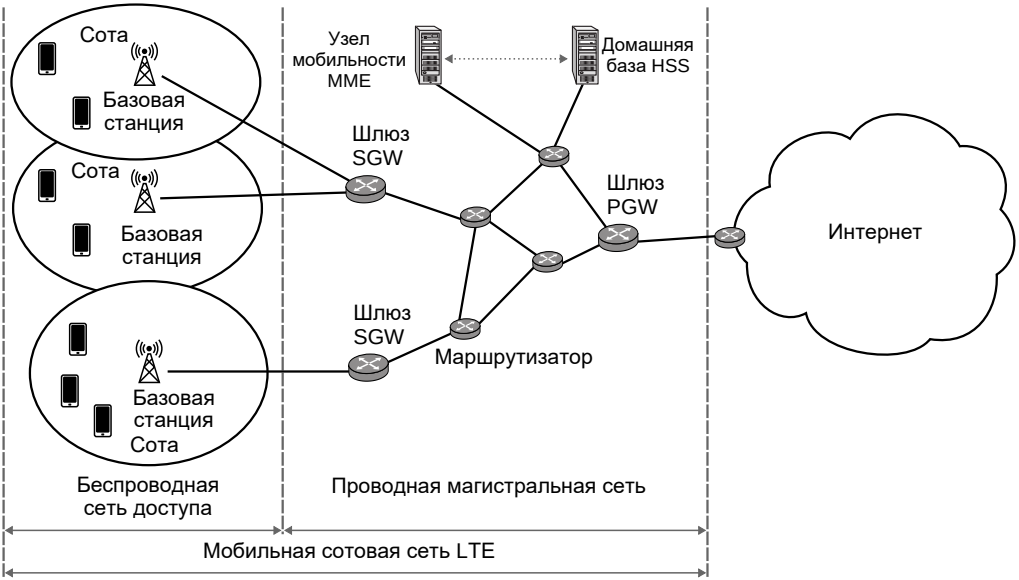


Рис. 24.1. Сотовая сеть провайдера

Очень важной является связь магистральной сети с интернетом. Через интернет в крупной сотовой сети могут осуществляться соединения различных географически отдаленных частей магистральной сети, через интернет взаимодействуют сотовые сети различных провайдеров, ну и, естественно, доступ к интернету является самой востребованной сегодня услугой сотовой сети.

Беспроводная сеть доступа

Сеть доступа образована большим количеством базовых станций, к которым подключены пользовательские устройства — телефоны.

Базовая станция — это беспроводной хаб (концентратор), который обеспечивает пользователей сети, находящихся в зоне покрытия базовой станции, доступом к магистральной сети. Базовая станция играет роль *пограничного устройства* между двумя видами сетей: с одной стороны она связана с беспроводными устройствами пользователей, а с другой — соединена проводной связью с магистральной сетью. Для связи с устройствами пользователей (телефонами) базовая станция оснащена мощными радиопередатчиками и чувствительными антеннами, обеспечивающими надежную радиосвязь в зоне покрытия. Базовая станция имеет обычный IP-адрес.

Сота — зона покрытия одной базовой станции. Она занимает сравнительно небольшую территорию — от сотни метров в городах до 10–35 км в сельской местности. В совокупности соты образуют **территорию покрытия мобильной сети**.

Базовая станция выполняет не только роль приемника и передатчика радиосигналов, но также *распределяет пропускную способность радиосреды* соты между телефонами пользователей, находящихся в данной соте. Пропускная способность радиосреды является дефицитным ресурсом, так как диапазон частот, выделяемый оператору мобильной сотовой сети государственным регулятором (сотовые мобильные сети работают в лицензированных, а следовательно, регулируемых диапазонах частот) ограничен. В мобильных сотовых сетях применяется *централизованный* метод доступа, который обеспечивает более эффективное использование полосы частот диапазона по сравнению с распределенным методом доступа.

Одна базовая станция может обслуживать и несколько сот, используя для создания каждой соты направленные антенны. В таком случае базовая станция использует для каждой своей соты разные диапазоны частот.

Для распределения пропускной способности радиосреды между телефонами пользователей в сотах базовая станция применяет различные методы частотного мультиплексирования, позволяющие в одной и той же разделяемой среде одновременно передавать потоки данных многих пользователей. Так как базовые станции управляют разделением частотных каналов радиосреды своих сот независимо друг от друга, то при использовании одного и того же диапазона частот во всех сотах сети может случиться так, что два пользователя из разных сот получат от своих базовых станций одну и ту же частоту. Если эти пользователи окажутся слишком близко друг к другу (на границах своих сот), то может произойти интерференция сигналов их телефонов, что ухудшит качество их связи. Значит, желательно назначать всем пользователям мобильной сети разные частоты, но понятно, что это расточительное решение — на всех частот не хватит. Поскольку интерференция сигналов одной частоты наиболее сильно проявляется между смежными сотами, то нужно по крайней мере сделать так, чтобы смежные соты работали в различных диапазонах частот. Так как диапазон частот нужно экономить, то в несмежных сотах можно использовать одну и ту же частоту — такая техника называется *повторным использованием частоты*.

На рис. 24.2 показан вариант организации сот при наличии всего трех частот, при этом ни одна из соседних пар сот не задействует одну и ту же частоту. Многократное использование частот позволяет избежать интерференции сигналов соседних сот, но при этом оператор может применить в каждой соте только часть диапазона частот, выделенных ему для работы сети. Например, при использовании трех повторяющихся диапазонов частот каждой соте достается только треть полного диапазона сети оператора, а значит, и количество одновременных соединений в соте уменьшается в три раза по сравнению со случаем, когда каждая сота пользуется полным диапазоном.

Некоторые современные виды частотного мультиплексирования позволяют смежным сотам работать на одной и той же частоте, при этом проблема интерференции сигналов пользователей смежных сот решается за счет динамического управления мощностью сигнала каждого телефона, направленными антеннами MIMO и некоторыми другими техническим приемами. В то же время техника многократного использования частот в сотах существенно упрощает решение проблемы интерференции сигналов соседних сот, и этот прием по-прежнему используется операторами современных мобильных сотовых сетей, особенно операторами сетей 5G в сотах небольших размеров, где интерференцию трудно компенсировать из-за близости базовых станций.

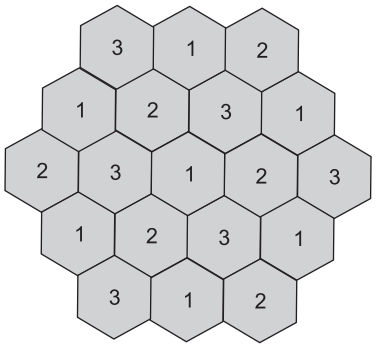


Рис. 24.2. Многократное использование частот в сотовой сети

Данные в радиосети передаются кадрами небольшого размера, которые можно быстрее восстановить при достаточно частых потерях и искажениях, характерных для беспроводной среды. Стек протоколов сети радиодоступа отличается от стека протоколов магистральной сети: если в магистрали работают стандартные протоколы проводной сети, то есть протокол Ethernet работает под протоколами стека TCP/IP, то в радиосети доступа на уровнях ниже IP применяется собственный стек протоколов беспроводной связи.

В радиосети доступа прямое взаимодействие устройств пользователя друг с другом не предусмотрено. Они могут обмениваться данными только с базовой станцией. А поскольку все устройства пользователя связаны с базовой станцией напрямую, то сеть радиодоступа является *одношаговой сетью* (single-hop), в которой нет необходимости маршрутизировать кадры.

Пользовательские данные надежно шифруются в беспроводной сети доступа, обеспечивая защиту от прослушивания и разрушения.

Магистральная проводная сеть

Магистральная сеть LTE — это проводная IP-сеть, построенная в соответствии со стандартной архитектурой глобальной IP-сети, то есть на основе IP-маршрутизаторов, а также коммутаторов Ethernet. В IP-пакетах передаются различные классы трафика — компьютерные данные различных приложений и данные голосовых телефонных сессий. Магистральная проводная сеть может иметь произвольную топологию, поэтому здесь необходимо применять маршрутизацию.

Маршрутизаторы, кроме выполнения своих стандартных функций по маршрутизации IP-пакетов и поддержанию качества обслуживания (QoS), выполняют ряд важных дополнительных действий, связанных с обеспечением мобильности абонентов сети. Эти функции они выполняют совместно с рядом серверов, которые помимо этого поддерживают аутентификацию пользователей, переадресацию звонков, функции автоответчика и т. п.

Основным узлом сети, отвечающим за поддержание мобильности устройств пользователей, является **сервер управления мобильностью MME** (Mobilty Managemet Entity). Этот сервер знает топологию сети — IP-адреса всех ее узлов. Его главная задача состоит в отслеживании перемещения устройств пользователя из соты в соту, об этих перемещениях ему

сообщают базовые станции. Сервер MME координирует изменения маршрутов передачи пакетов в сети в соответствии с этими перемещениями. Сервер MME управляет двумя специальными маршрутизаторами:

- **Пакетным шлюзом PGW** (Packet Data Network GateWay), являющимся пограничным маршрутизатором магистральной сети, соединяющим ее с интернетом.
- **Сервисным шлюзом SGW** (Serving GateWay), являющимся связующим звеном между пакетным шлюзом и базовыми станциями.

Сервер MME проводит также *аутентификацию* пользователя, обращаясь к серверу **HSS** (Home Subscriber Server) — **домашней базе данных провайдера сети**. В базе HSS хранятся учетные данные каждого пользователя сети, включая уникальный идентификатор, секретный ключ, номер (или номера) телефона, а также детали подписки на услуги сети.

Устройства пользователя

Устройство пользователя (User Equipment, **UE**) мобильной сотовой сети представляет собой специализированный компьютер (смартфон, планшет, настольный компьютер и т. п.), оснащенный радиоприемопередатчиком, антеннами и поддерживающий стек протоколов IP и LTE. Термин «устройство пользователя» отражает тот факт, что пользователи сетей LTE могут применять различные типы устройств для работы с сетью, а не только телефоны. В дальнейшем мы будем использовать термины «телефон» и «устройство пользователя» как синонимы. Устройство пользователя обычно имеет помимо радиоинтерфейса с сотовой сетью несколько радиоблоков (интерфейсов) для связи с сетями других беспроводных технологий, например Wi-FI, Bluetooth.

Устройства пользователя сетей LTE делятся на категории, в зависимости от характеристик их радиоблока и максимальной скорости передачи данных. *Устройства категории 1* — это самые простые устройства, имеют 1 антенну и способны поддерживать скорость передачи данных до 10 Мбит/с в нисходящем направлении (от базовой станции к телефону) и до 5 Мбит/с в восходящем направлении.

Устройства категории 20 — самые совершенные, они могут иметь несколько MIMO антенн в конфигурации 2 × 2, 4 × 4 или 8 × 8 и поддерживать скорость до 2 Гбит/с в нисходящем направлении и до 315 Мбит/с в восходящем. К этой категории в 2026 году относился, например, Samsung Galaxy S20 Ultra.

Для пользования услугами мобильной сети LTE в устройстве пользователя обязательно должна иметься сменная **SIM-карта** (Subscriber Identity Module, **SIM**). SIM-карта конфигурируется провайдером услуг LTE, это своего рода паспорт устройства пользователя, его пропуск для входа в сеть. В SIM-карте хранятся помимо прочих следующие данные:

- Уникальный **идентификатор абонента** (International Mobile Subscriber Identity, **IMSI**). Этот идентификатор сохраняется неизменным в течение всей жизни SIM-карты. Он используется при идентификации пользователя при подключении к сети.
- Отдельные уникальные публичный и частный **идентификаторы абонента мультимедийных услуг**. Эти идентификаторы используются при регистрации пользователя мультимедийных услуг.
- **Секретный ключ**, используемый для аутентификации абонента и шифрации его данных.

Копии этих данных хранятся в HSS — домашней базе данных пользователей провайдера услуг сети LTE, поэтому пользователь освобожден от необходимости вводить пароль каждый раз, когда его телефон переходит в активное состояние и присоединяется к сети, за него это делает сам телефон, используя информацию SIM-карты. Заметим, что номер телефона в SIM-карте на хранится: он имеется только в базе данных провайдера. В SIM-карте не хранится и IP-адрес телефона: он назначается телефону сетью динамически. Идентификатор IMSI является уникальным кодом, его можно сравнить с MAC-адресом, уникально определяющим узел сети на канальном уровне. С одним и тем же идентификатором IMSI может быть связано *несколько* телефонных номеров.

Туннелирование

Мобильность — основная цель сотовых сетей

Как уже было сказано, современные сотовые сети являются IP-сетями. Все узлы сотовой сети (маршрутизаторы, серверы, клиентские станции, базовые станции) и конечные узлы (телефоны) имеют IP-адреса. Однако, как мы покажем ниже, механический перенос и опора на использование исключительно традиционной технологии IP-сетей для решения задач сотовой связи вряд ли возможны. И главным препятствием этому является требование *мобильности* телефонов.

Мы уже встречались с проблемой мобильности в IP-сетях, которая заключается в том, что когда узел перемещается из одной IP-сети в другую, он либо сохраняет старый адрес, и тогда возникают сложности с маршрутизацией трафика, направляемого по этому адресу. Либо узел получает новый IP-адрес, и тогда все прикладные сессии, начатые узлом к моменту перемещения, прерываются из-за несоответствия сокетов (IP-адрес, номер порта) новому IP-адресу.

Решением этой проблемы в IP-сетях является технология *Мобильный IP* (читайте о ней в главе 14). При разработке этой технологии предполагалось, что большинство узлов сети являются стационарными, а те, которые перемещаются, делают это эпизодически. Однако это условие абсолютно не соответствует картине, которая характерна для сотовых сетей: здесь, когда узлы-телефоны меняют точку присоединения к сети, то:

- они делают это часто;
- они делают это быстро;
- они делают это в массовом порядке — таких узлов сотни и тысячи у мобильной сети провайдера.

Мобильность в сотовых сетях является *фундаментальным требованием, главной целью* создания сети. Эта цель состоит в том, чтобы предоставить пользователям сети полную свободу перемещения со своими телефонами так, чтобы они не замечали переключения с соты на соту и продолжали работу с любыми сетевыми приложениями без прерываний. Именно исходя из этих требований была разработана специальная технология сотовых сетей, учитывающая специфические особенности и требования, нацеленные на поддержку мобильности массового перемещения узлов.

Одним из основных средств достижения мобильности в сотовых сетях является **туннелирование**.

Схема перемещения пакета по сотовой сети

Мы начнем изучение механизмов туннелирования в сотовых сетях с рассмотрения очень упрощенной схемы (рис. 24.3), на которой показано перемещение IP-пакета от веб-сайта, находящегося в интернете, к телефону, присоединенному к мобильной сотовой сети.

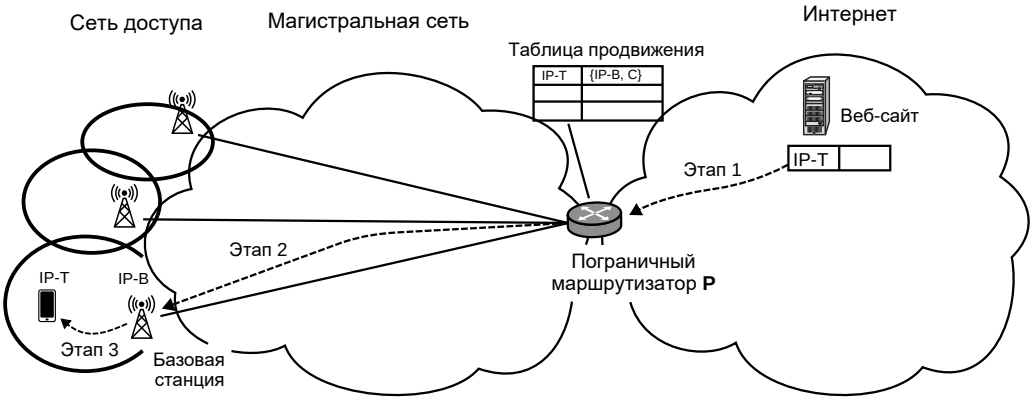


Рис. 24.3. Упрощенная схема передачи пакета телефону

Магистральная сеть провайдера соединена с интернетом через пограничный маршрутизатор P. Базовые станции соединены с пограничным маршрутизатором P промежуточными IP-маршрутизаторами, не показанными на рисунке для сохранения его ясности. Все узлы сотовой сети имеют обычные IP-адреса. IP-адреса назначаются телефонам независимо от их точек присоединения к базовым станциям и IP-адресов других узлов сети.

Глядя на топологию сети доступа, нетрудно заметить, что если в обычной IP-сети IP-адреса узлов отражают структуру сети, полученную путем разбиения сети на подсети, то использование IP-адресов для адресации телефонов никак не отражает сотовую структуру сети доступа. Интуитивно понятной системой адресации в данном случае оказывается адресация телефонов в виде пары {*идентификатор соты, локальный идентификатор телефона в соте*}.

В качестве идентификатора соты можно использовать IP-адрес базовой станции, а в качестве локального идентификатора телефона можно использовать уникальный в пределах соты идентификатор, который назначается телефону базовой станцией при его присоединении к соте. Назовем адрес телефона, заданный парой {IP-адрес базовой станции; локальный идентификатор телефона} — *«сотовым адресом»*. (Подчеркнем, что этот адрес введен исключительно в учебных целях для пояснения механизма мобильности в данной упрощенной схеме и не имеет прямого аналога в реальной сотовой сети.)

Пусть телефон, показанный на рис. 24.3, имеет IP-адрес, равный IP-T. Этот телефон подключен к базовой станции, имеющей адрес IP-B, то есть его «сотовый адрес» — {IP-B, C}, где C — локальный идентификатор телефона.

Использование двух разных систем адресации телефонов — IP-адресов и «сотовых адресов» — позволяет решить дилемму мобильности IP-сетей.

ДИЛЕММА МОБИЛЬНОСТИ IP-СЕТЕЙ

Либо оставлять IP-адрес узла неизменным при его перемещении в другую сеть, тогда сохраняются активные сессии приложений, но возникают сложности с маршрутизацией трафика.

Либо принять новый адрес в гостевой сети и тем самым упростить маршрутизацию, но тогда разрываются сессии приложений.

Теперь каждый тип адреса телефона выполняет свою функцию.

- *«Сотовый адрес»* меняется при каждом переходе из соты в соту (меняется и адрес базовой станции, и локальный идентификатор телефона), показывая, где, в какой соте находится в данный момент телефон, куда надо направлять адресованный ему трафик.
- *IP-адрес* остается неизменным при перемещении телефона в другую соту, значения сокетов приложений не меняются, и следовательно, прикладные сессии не прерываются.

Рассмотрим упрощенную схему передачи пакетов от узла интернета на телефон, имеющий глобальный IP-адрес IP-T и локальный «сотовый адрес» {IP-B, C}. Весь путь разбивается на два этапа.

Первый этап — перемещение пакета от внешнего узла интернета до пограничного маршрутизатора магистральной сети провайдера сотовой сети. На этом этапе данные продвигаются на основании IP-адреса телефона.

Пограничный маршрутизатор P рутинным образом объявляет своим соседям по интернету обо всех IP-подсетях сети провайдера, в том числе и о сети, к которой принадлежит адрес телефона, поэтому маршруты ко всем узлам сети провайдера через интернет известны. Пакет с адресом IP-T, который веб-сайт интернета отправляет телефону, маршрутизируется обычным образом, пока не достигнет пограничного маршрутизатора сети провайдера.

Второй этап — доставка пакета от пограничного маршрутизатора до телефона пользователя. На этом этапе для направления трафика вместо IP-адреса телефона (IP-T) начинает использоваться его «сотовый адрес» {IP-B, C}.

- На пограничном маршрутизаторе конфигурируется запись специальной таблицы продвижения пакетов через магистральную сеть. Эта запись отображает адрес телефона IP-T на «сотовый адрес» {IP-B, C} (запись показана на рис. 24.3).
- Все пакеты, адресованные телефонам, присоединенным к мобильной сотовой сети, приходят на пограничный маршрутизатор P, так как именно он объявляет адрес своего внешнего интерфейса как входной точки для всех IP-префиксов сетей, входящих в мобильную сеть провайдера.
- Пограничный маршрутизатор проверяет свою специальную таблицу продвижения и, если находит совпадение IP-адреса назначения пакета с адресом телефона IP-T в таблице, то обрабатывает пришедший пакет специальным образом. Пакет инкапсулируется в другой, внешний пакет, адресом назначения становится адрес базовой станции IP-B из записи таблицы продвижения, а локальный идентификатор телефона C помещается в специальное поле внешнего пакета.
- Внешний пакет отправляется пограничным маршрутизатором P в магистральную сеть, и ее промежуточные маршрутизаторы с помощью стандартной маршрутизации

доставляют этот пакет базовой станции с адресом IP-B. Внутреннего исходного пакета промежуточные маршрутизаторы не замечают.

- Базовая станция получает внешний пакет и извлекает (декапсулирует) из него исходный пакет, а также читает из специального поля внешнего пакета значение локального идентификатора C.
- Базовая станция передает исходный пакет телефону, но не на основе его IP-адреса, а на основе локального идентификатора C.

Описанная схема решает поставленную задачу, пакет доставлен телефону через магистральную сеть без использования IP-адреса телефона, на основании его «сотового адреса». Пограничный маршрутизатор P для реализации такой схемы должен знать отображение IP-адреса телефона на его «сотовый адрес». Мобильность телефона в нашей упрощенной схеме поддерживается на основе обновления записи отображения при каждом перемещении телефона между сотами. Для этого базовая станция и пограничный маршрутизатор должны поддерживать между собой логическое соединение, в рамках которого базовая станция сообщает пограничному маршрутизатору новый «сотовый адрес» телефона, при его присоединении к соте этой базовой станции.

Нетрудно заметить, что в результате действий, выполненных в рамках второго этапа, между пограничным маршрутизатором и базовой станцией был установлен **транспортный туннель**[1]. Действительно, описанный способ передачи удовлетворяет всем признакам передачи трафика по туннелю:

- существуют два узла, являющиеся конечными точками туннеля;
- имеется критерий, в соответствии с которым все пакеты, удовлетворяющие этому критерию, инкапсулируются во внешний пакет и направляются в туннель;
- по туннелю исходные пакеты передаются как пассажиры и на маршрутизацию в туннеле они не влияют;
- на выходе из туннеля пакеты декапсулируются и в исходном виде направляются далее в сеть;
- конечные точки обмениваются служебной информацией в рамках логического соединения (служебного протокола) для установления специальных таблиц продвижения, на основе которых выполняется инкапсуляция и декапсуляция пакетов.

Двухуровневая система туннелей

Описанная выше упрощенная схема поддерживает мобильность телефонов в сети, но она плохо масштабируется, потому что пограничный маршрутизатор должен выполнять слишком большой объем работы. К его обычным обязанностям по маршрутизации всего трафика между телефонами и интернетом добавилась работа по созданию новых туннелей при каждом переходе телефона из соты в соту. Для того чтобы перенести часть этой работы на другие маршрутизаторы, в сети LTE используется иерархическая двухуровневая система туннелей (рис. 24.4).

---

[1] Читайте о туннелях в главах 14 (раздел «Мобильный IPv4») и 17 (раздел «Процесс адаптации версии IP6»).

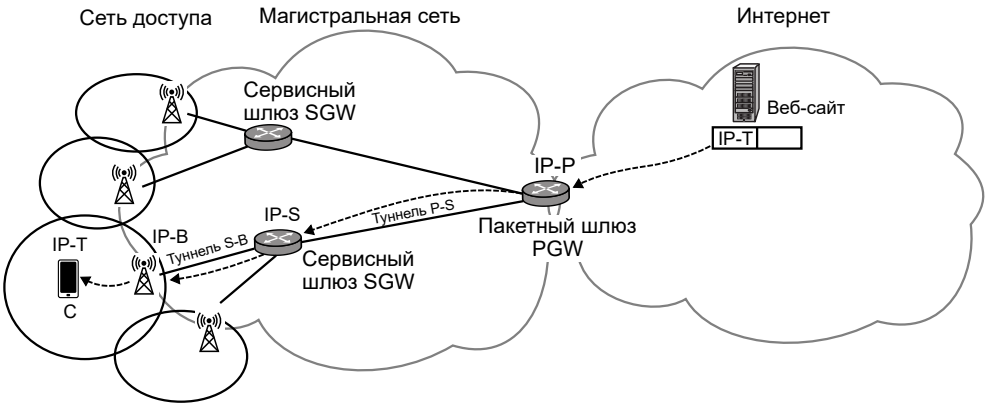


Рис. 24.4. Двухуровневая система туннелей

В помощь пограничному маршрутизатору, роль которого в сети LTE играет пакетный шлюз PGW, привлекается второй уровень маршрутизаторов — сервисные шлюзы SGW. Такая схема позволяет для каждого соединения *базовая станция — пограничный маршрутизатор* разделить один туннель на два:

- туннель P-S между пакетным шлюзом PGW и сервисный шлюзом SGW и
- туннель S-B между сервисным шлюзом SGW и базовой станцией.

Как и в предыдущей схеме, исходные пакеты перемещаются по туннелям в инкапсулированном виде. Адресом назначения *внешнего* пакета в туннеле P-S теперь является адрес сервисного шлюза IP-S, а в туннеле S-B — адрес базовой станции IP-B.

Сервисный шлюз SGW совмещает две роли: конечной точки туннеля P-S и начальной точки туннеля S-B. Соответственно, маршрутизатор S должен извлечь исходный пакет из внешнего пакета с адресом назначения IP-S и заново инкапсулировать его в новый внешний пакет с адресом назначения IP-B.

Такая двухшаговая древовидная организация туннелей позволяет более эффективно поддерживать *мобильность* пользователей в сотовых сетях. Действительно, обычно сервисные маршрутизаторы SGW (а в крупной сети их может быть много) географически распределены так, что каждый маршрутизатор SGW обслуживает базовые станции одного географического района, например Новосибирска или Барнаула. Так как телефон пользователя будет перемещаться в течение звонка или сессии видеостриминга, скорее всего, между базовыми станциями, расположенными в этом районе (даже если он едет на машине или поезде), то переход из одной соты в другую будет происходить путем изменения только туннелей S-B, а туннель P-S станется неизменным. Такая организация туннелей разгружает пакетный шлюз PGW от большей части работ по поддержанию мобильности, которая переносится на сервисные шлюзы SGW и базовые станции.

При двухуровневой организации туннелей конечной точкой туннеля, в котором задействован пограничный (пакетный шлюз PGW) маршрутизатор, является не базовая станция, как это было в упрощенной схеме, а сервисный шлюз SGW. Поэтому пакетному шлюзу PGW нет смысла использовать в своей таблице продвижения «сотовый адрес» телефона, то есть пару (IP-адрес базовой станции, локальный идентификатор телефона), вместо этого ему требуется отображение IP-адреса телефона на IP-адрес соединенного с ним сервисного

шлюза сети. Использование пакетным шлюзом локального идентификатора телефона тоже нежелательно, так как при перемещении телефона из соты в соту пакетному шлюзу пришлось бы перестраивать запись таблицы продвижения, соответствующую данному телефону, а это плохо масштабируемое решение, желательно локальный идентификатор телефона использовать только на последнем этапе — при передаче IP-пакета базовой станцией телефону.

А если локальный идентификатор не используется пакетным шлюзом PGW при туннелировании исходного пакета, то каким образом передать базовой станции информацию о том, какому телефону предназначен дошедший до станции по составному туннелю пакет? Каким образом можно связать между собой туннели так, чтобы исходный пакет в конце концов дошел до нужной базовой станции, а потом и до телефона? В сотовых сетях предлагается решение, использующее более быстрый по сравнению с IP-маршрутизацией способ продвижения пакета — *метод коммутации меток*, применяемый в технике виртуальных каналов (см. главу 20).

Метод коммутации меток требует предварительного формирования таблиц коммутации. Для того чтобы понять, как они формируются и как конфигурируются туннели, рассмотрим более детально механизм туннелей сетей LTE.

Туннель GTP

Сети LTE унаследовали технику туннелирования от второго поколения сетей GPRS, что следует из названия туннелей — GTP. Эта аббревиатура означает GPRS Tunneling Protocol.

Туннель GTP переносит пользовательские IP-пакеты внутри сообщений протокола UDP. Туннель образуется между двумя узлами IP-сети, называемыми часто шлюзами, и состоит из двух однонаправленных путей трафика. Туннель образуется в результате установления *логического соединения* между шлюзами туннеля. В рамках этого соединения узлы обмениваются параметрами туннеля. Конечные точки туннеля соединены обычной IP-сетью, через которую эти точки должны быть достижимы.

Рисунок 24.5 иллюстрирует создание туннеля GTP между двумя шлюзами-маршрутизаторами, *A* и *B*. Каждый однонаправленный путь в шлюзе идентифицируется парой значений:

- IP-адресом конечной точки туннеля для данного направления и
- идентификатором туннеля TEID также для данного направления.

Идентификатор TEID (Tunnel Endpoint ID) длиной 32 бита имеет локальное значение для каждой пары шлюзов, между которыми устанавливается туннель.

Способ идентификации туннеля GTP рассчитан на то, что между двумя узлами сети может существовать несколько туннелей и один туннель может быть связан с другим туннелем так, что конечная точка одного туннеля является начальной точкой другого туннеля (на рисунке шлюз *A* соединяет несколько туннелей). Идентификация туннеля парой значений {IP, TEID} позволяет различать туннели в системе туннелей и связывать их в последовательность для образования некоторого маршрута через сеть.

Значения идентификаторов TEID выбираются шлюзами туннеля, при этом для каждого направления туннеля выбор идентификатора делается шлюзом, который *принимает* трафик. Этот шлюз говорит своему визави: «Для туннеля, который мы устанавливаем, помечай пакеты идентификатором со значением TEID=N. Тогда я буду понимать, что это пакеты данного туннеля, а не какого-то другого, который, возможно, существует между нами».

В нашем примере направление $B \rightarrow A$ идентифицируется парой {132.13.1.3, 12008}, где первое значение — это IP-адрес шлюза *A* (обозначим его IP-*A*), а второе — значение идентификатора TEID=12008, которое шлюз *A* (конечная точка направления $B \rightarrow A$) выбрал и сообщил в процессе установления туннеля шлюзу *B*. Противоположное направление $A \rightarrow B$ идентифицируется парой {132.14.32.6, 12007}, где значение идентификатора TEID=12007 выбрал узел *B*.

Пусть на внешний интерфейс пограничного шлюза *B* поступает трафик для телефона с IP-адресом 191.45.2.5, обозначим его IP-T (см. на рисунке исходный IP-пакет).

У каждого шлюза имеется таблица отображения входного трафика на туннель, которая говорит шлюзу, пакеты какого потока нужно направлять в туннель, а какие нужно передавать с обычной логикой IP-протокола. В простейшем случае поток определяется одним признаком — IP-адресом назначения, но в общем случае поток можно определять по многим признакам: IP-адресам источника и назначения, TCP/UDP-портам, классу трафика и метке потока IPv6.

На рис. 24.5 показана одна запись таблицы отображения шлюза *B*. Она говорит ему, что пакеты, поступающие на его входной интерфейс с адресом назначения 191.45.2.5, нужно направлять в туннель, определяемый парой значений {132.13.1.3, 12008}.

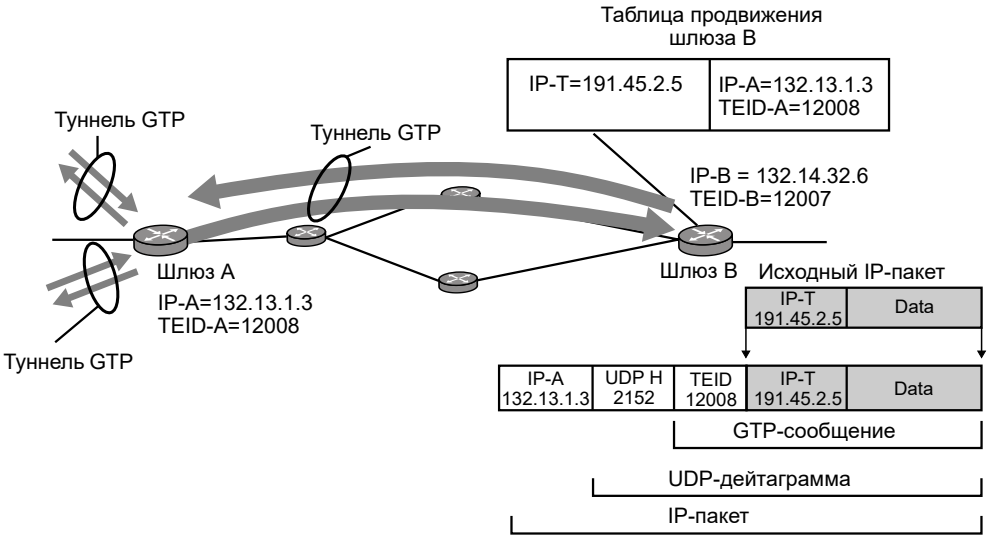


Рис. 24.5. Туннель GTP

На рисунке показан процесс обработки шлюзом *B* пришедшего пакета с адресом назначения IP-T = 191.45.2.5. Так как этот адрес совпадает с признаком потока для туннелирования, то шлюз B создает новый IP-пакет с адресом назначения из таблицы отображения — IP-*A* = 132.13.1.3. В поле заголовка протокола UDP записывается известный номер порта 2152, который закреплен за протоколом GTP. В поле метки TEID заголовка протокола GTP помещается значение 12008, а в поле данных сообщения протокола GTP помещается туннелируемый IP-пакет.

Сформированный таким образом новый IP-пакет отправляется в магистральную сеть. Этот пакет маршрутизируется и продвигается обычным образом по адресу назначения

IP-*A* = 132.13.1.3. При поступлении пакета на шлюз *A* он распознается как пакет туннеля (по известному UDP-порту) и декапсулируется, то есть из поля данных дейтаграммы UDP извлекается исходный пакет с адресом назначения IP-T=191.45.2.5 и *идентификатор туннеля TEID*.

Если шлюз *A* является пограничным шлюзом сети, как и шлюз *B*, то оригинальный пакет отправляется далее уже без туннелирования, так как система туннелей на шлюзе *A* закончилась. Если же от шлюза *A* ответвляются другие туннели, как это показано на рис. 24.5, то оригинальный пакет снова инкапсулируется по правилам протокола GTP и направляется в один из следующих туннелей. Для выбора следующего туннеля служит идентификатор туннеля TEID. Как этот выбор осуществляется, мы увидим чуть позже, при рассмотрении работы системы туннелей магистральной сети LTE.

Маршрут инкапсулированного пакета по туннелю между его конечными точками, то есть последовательность промежуточных маршрутизаторов, через которые проходит трафик туннеля, определяется протоколом маршрутизации, который работает в магистральной сети. Для целей туннелирования этот маршрут не имеет значения. Важно только то, чтобы все пакеты, предназначенные для туннелирования, входили в одну его конечную точку и выходили из другой конечной точки.

На шлюзах должно быть установлено дополнительное программное обеспечение, которое управляет созданием туннелей и продвижением пакетов через него. Служебный протокол, который используют конечные узлы для создания туннелей, называется GTP-C (GTP-Control, GTP управления), а протокол продвижения пользовательских данных через туннели — GTP-U (GTP-User, пользовательский GTP).

На рис. 24.6 показано использование туннелей GTP в магистрали LTE. Этот рисунок конкретизирует схему, представленную на рис. 24.1. Связи протоколов управления отражают тот факт, что туннели в сети LTE прокладываются при централизованном контроле

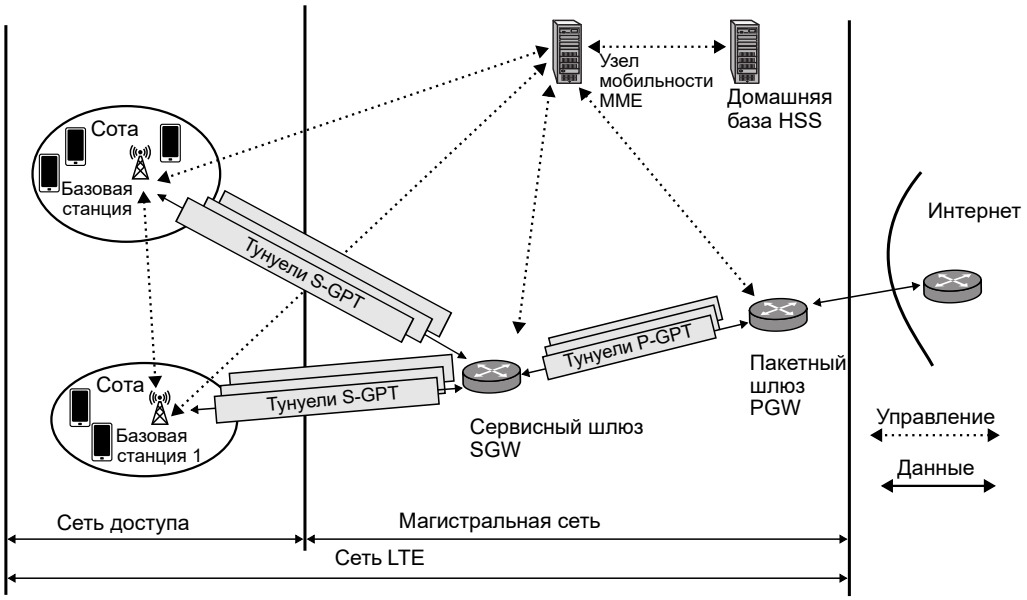


Рис. 24.6. Применение туннелей GTP в магистральной сети

со стороны сервера управления мобильностью. Этот сервер получает от базовых станций информацию о перемещении телефонов из соты в соту и координирует работу по изменению туннелей S-B для образования новых маршрутов для изменивших свои точки подключения телефонов. (В стандартах технологии LTE вместо используемых нами для простоты названий туннелей S-B и P-S используются названия S5/S8 GTP и S1 GTP соответственно.)

Процесс создания таблиц продвижения шлюзов PGW и SGW рассмотрен далее, при описании процедуры присоединения телефона к сети.

Присоединение устройства пользователя к сети

Первой операцией, которую выполняет устройство пользователя при включении питания, является присоединение к мобильной сети. В этой операции участвуют многие узлы и серверы сети, представленные на рис. 24.1. Присоединение к сети происходит в несколько этапов:

1. Присоединение к соте.
2. Аутентификация пользователя.
3. Создание туннелей для передачи трафика пользователя через магистраль.

Этап 1. Присоединение к соте

После включения питания устройство пользователя начинает прослушивать частотные диапазоны, отведенные сетям LTE. В этих диапазонах базовые станции передают синхросигналы, услышав которые телефон понимает, что он находится в зоне покрытия определенной соты. Каждая сота имеет свой идентификатор, который она сообщает в своих синхросигналах.

Если обнаружены сигналы нескольких сот, то устройство пользователя выбирает соту с наиболее мощным и стабильным сигналом. Обычно устройство пользователя хранит информацию о последней соте, к которой оно было присоединено до отключения питания, и старается найти ее синхросигналы, но если не находит, то выполняет поиск новой соты. Выбрав соту по лучшему сигналу, устройство пользователя использует ее сигналы для вхождения в посимвольную и покадровую синхронизацию, после чего заявляет о своем присутствии и желании присоединиться к соте отправкой своего первого сообщения — *преамбулы*. Преамбула отправляется по отдельному разделяемому частотному каналу, где она может столкнуться с преамбулой другого устройства, также присоединяющегося к сети.

Если преамбула была распознана базовой станцией (то есть коллизии не произошло), то она сообщает устройству пользователя его **временный сотовый идентификатор радиосети C-RNTI** (Cell Radio Network Temporary Identifier), который однозначно адресует телефон в соте. Этот идентификатор играет важную роль во взаимодействии базовой станции и устройства пользователя. Он используется базовой станцией как адрес канального уровня устройства пользователя, с его помощью базовая станция «помечает» данные, предназначенные тому или иному телефону, а телефон распознает, есть ли среди посланных в разделяемую среду данных такие, которые адресованы ему. Ранее мы обозначали этот

идентификатор как идентификатор C в целях сохранения ясности, чтобы не отвлекать читателя на запоминание не очень понятных аббревиатур.

В сетях LTE данные в радиосреде передаются в ресурсных блоках OFDMA (см. главу 23). Ресурсные блоки адресуются номерами, соответствующими положению блока в частотном диапазоне сети — первый, второй и т. д. Так что устройству пользователя нужно только знать номер или номера ресурсных блоков, в котором (которых) ему передали данные в очередном тайм-слоте.

Механизм применения идентификатора C-RNTI в качестве адреса весьма необычен для проводных сетей, поэтому рассмотрим его подробно.

Базовая станция помещает информацию о принадлежности ресурсных блоков устройствам пользователей в отведенные для этой цели начальные управляющие блоки переданного кадра. Эти служебные поля называются «поля DCI» (рис. 24.7). В каждом из них содержатся несколько номеров ресурсных блоков, адресованных определенному телефону. Адреса телефона в этом поле нет. Возникает вопрос, как телефон может понять, что в одном из служебных полей DCI находятся номера его ресурсных блоков? Для решения этой проблемы используется шифрация. Базовая станция вычисляет контрольную сумму содержимого поля DCI, несущего информацию о номерах ресурсных блоков для определенного телефона, и зашифровывает ее с помощью *временного сотового идентификатора* C-RNTI этого телефона, пользуясь им как ключом шифрования, а затем помещает зашифрованную контрольную сумму в поле DCI. В данном примере для телефона со значением идентификатора C-RNTI = 125 в поле DCI помещена информация о том, что в кадре есть ресурсные блоки для этого телефона — их номера 4 и 5.

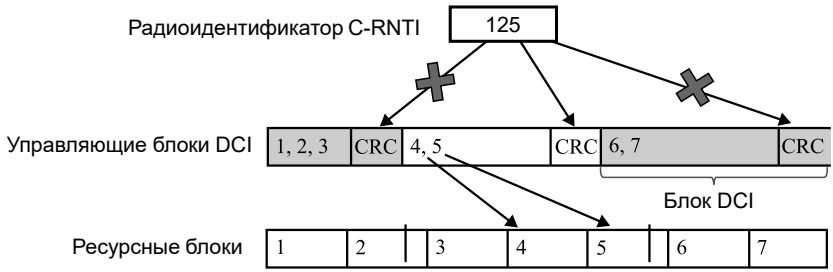


Рис. 24.7. Использование радиоидентификатора как адреса

При получении кадра каждый телефон считывает все служебные поля DCI, вычисляет контрольную сумму каждого из них и также зашифровывает ее с помощью своего временного сотового идентификатора. Если результат совпадает с помещенной в поле зашифрованной базовой станцией контрольной суммой, то значит, это поле предназначено данному телефону и он читает свои данные из ресурсных блоков кадра, номера которых он нашел в его служебном поле DCI. Возможно, что телефон не смог правильно расшифровать ни одно из служебных полей DCI, и это означает, что в текущем кадре для него данных нет. Рассматриваемый нами телефон оказался в числе тех, для которых в текущем кадре есть информация. Телефон, используя свой C-RNTI =12, смог найти совпадение контрольной суммы в одном из блоков DCI и узнал, что для него отведены ресурсные блоки 4 и 5. Как видно из описания, временный сотовый идентификатор C-RNTI является разделяемым секретом базовой станции и устройства пользователя.

Затем устройство запрашивает у станции разрешение на получение ресурсов радиосреды соты. Базовая станция анализирует загрузку соты и, если она позволяет присоединить к ней еще одно устройство, то отвечает положительно и выделяет устройству канал для отправки сигнальных сообщений. На этом процесс присоединения к соте заканчивается, и устройство переходит в состояние активности.

Этап 2. Аутентификация пользователя

Этот этап начинается с отсылки устройством пользователя запроса на присоединение к сети серверу управления мобильностью MME.

В этом запросе устройство указывает свой уникальный идентификатор IMSI, а также имя сети, к которой устройство хочет присоединиться через сеть LTE. Чаще всего это интернет.

Узел MME обращается к домашней базе данных HSS для проведения взаимной аутентификации пользователя и сети, передавая в качестве аргумента идентификатор пользователя IMSI. Узел HSS по идентификатору IMSI находит секретный ключ пользователя и с его помощью вырабатывает так называемый аутентификационный вектор, включающий несколько числовых значений, необходимых для процесса сильной аутентификации, в том числе ожидаемый ответ пользователя на слово-вызов, а также выработки *сессионных ключей* (техника сильной аутентификации рассматривается в главе 29). Сессионные ключи хранятся в устройстве пользователя и базовой станции, к которой он присоединен. Они используются для шифрации трафика пользователя в радиосети.

Вместе с этим узел MME уведомляет домашнюю базу данных о том, что устройство пользователя присоединено к сети в зоне ответственности данного узла MME, так что при необходимости связаться с этим устройством нужно обращаться к данному узлу MME. Сам узел MME запоминает в процессе присоединения, что устройство пользователя достижимо через базовую станцию. Убедившись, что проверка аутентичности пользователя прошла успешно, узел MME переходит к третьему этапу присоединения устройства пользователя к сети — этапу создания для данного устройства пользователя индивидуальных туннелей для его трафика.

Этап 3. Создание туннелей

Рассмотрим пример создания туннелей для присоединяющегося к сети устройства пользователя/телефона T1 (рис. 24.8).

Создание туннелей инициирует сервер мобильности MME. Он это делает сразу же после успешной аутентификации пользователя телефона T1. К этому моменту серверу MME известны:

- глобальный идентификатор устройства IMCI;
- факт присоединения телефона T1 к базовой станции 1;
- тип услуги, которую телефон запросил у сети (пусть это будет услуга сети передачи данных).

Телефон и связанная с ним базовая станция знают временный локальный идентификатор C-RNTI, назначенный телефону T1.

Действия по созданию туннелей могут быть разбиты на несколько шагов.

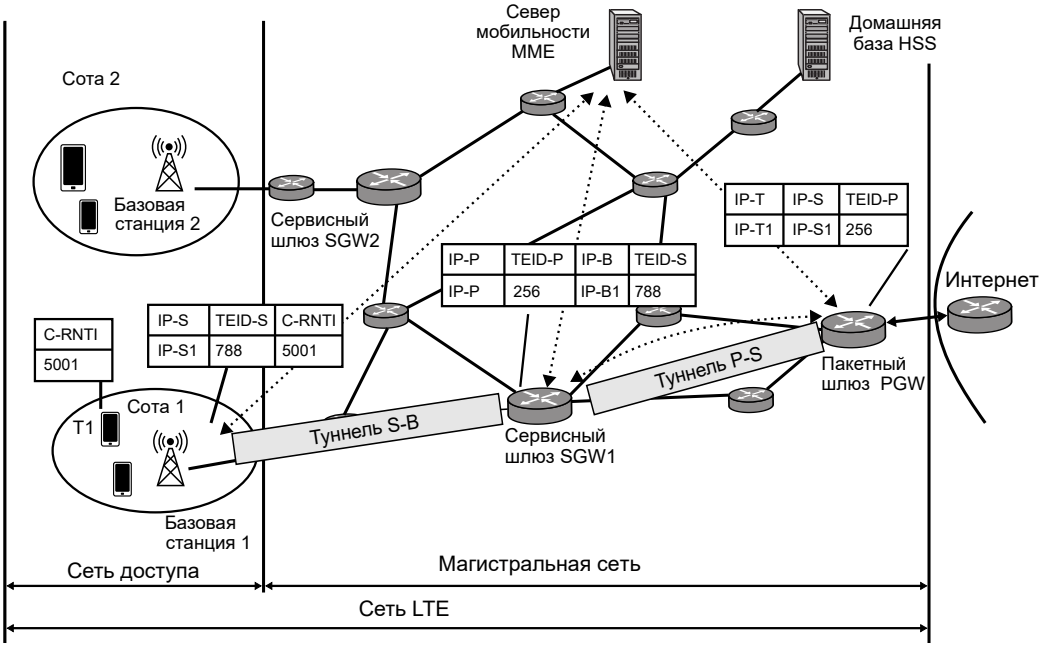


Рис. 24.8. Процесс конфигурирования туннелей магистральной сети

Начальным шагом является выбор сервером MME маршрута через магистральную сеть для соединения телефона T1 с пакетным шлюзом PGW. Этот маршрут будет обслуживать запрошенное телефоном T1 в процессе аутентификации соединение с интернетом (услуга сети передачи данных). Между телефоном и пакетным шлюзом могут существовать в общем случае несколько маршрутов. Пусть сервер управления мобильности выбрал маршрут:

T1 — Базовая станция 1- SGW1 — PGW

Этот маршрут нужно проложить в сети, то есть создать два туннеля и соединить их между собой.

Первым создается туннель P-S. Сервер управления мобильностью MME посылает сервисному шлюзу SGW1 запрос о необходимости создания туннеля P-S для трафика телефона T1. При этом в запросе указывается глобальный идентификатор телефона IMSI.

Шлюз SGW1 начинает выполнять этот запрос. Он пересылает запрос о создании туннеля шлюзу PGW, сообщая в нем глобальный идентификатор телефона T1 и значение выбранного им идентификатора создаваемого туннеля TEID-P для направления P → S (напомним, что идентификатор TEID назначает принимающий узел направления), в нашем примере оно равно 256.

Шлюз PGW *назначает IP-адрес* телефону T1, он обозначен в примере как IP-T1, и теперь шлюз PGW может создать для трафика телефона T1 запись в таблице продвижения: IP-T1 → IP-S1, 256 (рис. 24.8).

Шлюз PGW сообщает серверу MME значение назначенного телефону IP-адреса, и сервер MME передает его через сигнальный протокол телефону T1 (транзитом через базовую станцию). *Теперь у телефона есть IP-адрес,* и он может использовать его для обмена данными с любым узлом интернета — но только после завершения создания туннелей.

Шлюз PGW передает шлюзу SGW1 значение идентификатора TEID-S для направления S → P, шлюз SGW1 будет его использовать при отправке пакетов телефона T1 шлюзу PGW, то есть в обратном направлении туннеля P-S. На этом создание туннеля P-S завершено, и оба шлюза создали для него таблицы отображения трафика (на рисунке показаны таблицы только для направления «шлюз PGW → устройство пользователя T1»).

Похожим образом создается туннель S-B. Этот процесс также координирует сервер MME.

В результате согласованной работы трех транспортных узлов сети — пакетного шлюза PGW, сервисного шлюза SGW1 и базовой станции 1 — при координации всех операций севером управления мобильностью MME создан составной туннель, обеспечивающий обмен трафиком устройства T1 и шлюза PGW через магистральную сеть без использования IP-адреса устройства. Этот адрес учитывается только при приходе IP-пакетов для устройства T1 из интернета на входной интерфейс пакетного шлюза PGW.

Продвижение пакета через сеть

Посмотрим по шагам, как работает созданный составной туннель, когда по нему передается пакет, поступивший из интернета. Назовем его пакетом P.

1. Шлюз PGW сравнивает IP-адрес пришедшего пакета с записями своей таблицы продвижения. Он находит запись для адреса IP-T1, которая показывает, что пакет нужно направить в туннель {IP-S, 256}. Шлюз PGW инкапсулирует пришедший пакет в сообщение IP/UDP/GTP (в соответствии с правилами, приведенными выше) и помещает в соответствующие поля заголовков адрес IP-S1 и идентификатор TEID-P, равный 256, а пакет P — в поле данных протокола GPT.
2. Новый пакет, назовем его PP, направляется в магистральную сеть, маршрутизаторы которой доставляют его шлюзу SGW1, так как его адрес IP-S1 указан в качестве IP-адреса назначения нового пакета. Внутрь пакета промежуточные маршрутизаторы не заглядывают — для них это обычная процедура маршрутизации.
3. Шлюз SGW1 получает пакет PP в качестве хоста, а не маршрутизатора, потому что пакет адресован ему. По номеру UDP-порта 2151 шлюз вызывает протокол прикладного уровня, закрепленный за этим портом, которым является протокол GTP. Этот протокол распознает, по какому туннелю пришел к нему пакет, читая адрес отправителя из заголовка внешнего пакета (это IP-P) и идентификатор TEID-P=256 из поля TEID протокола GTP. Далее протокол GTP находит запись в своей таблице продвижения, содержащую пару {IP-P, TEID-P=256}, и узнает из этой записи, что пакет P нужно передать в туннель S-B идентифицируемой парой {IP-B1, TEID-S=788}. Для этого протокол формирует новый внешний пакет PS и инкапсулирует в него пакет P, указывая в качестве внешнего IP-адреса назначения IP-адрес базовой станции IP-B1 и в качестве идентификатора туннеля идентификатор TEID-S, равный 788.
4. Новый пакет PS направляется шлюзом SGW1 в сеть. Промежуточные маршрутизаторы доставляют его базовой станции 1, руководствуясь ее IP-адресом во внешнем пакете PS.

5. Базовая станция получает пакет PS, извлекает из него пакет P по правилам протокола GTP и использует пару {IP-S, TEID-S} для выбора нужной записи из своей таблицы продвижения. Эта запись говорит о том, что пакет должен быть передан телефону с локальным идентификатором C-RNTI=5001.
6. Базовая станция передает пакет телефону T1 в ресурсных блоках OFDMA по радиосреде, используя механизм адресации на основе ключа шифрования. Как было сказано выше, ключ шифрования совпадает с временным локальным идентификатором C-RNTI=5001. На этом путь пакета по составному туннелю завершился.

В обратном направлении процесс передачи пакетов от телефона T1 в интернет через составной туннель происходит аналогичным образом.

Можно заметить, что техника передачи трафика по составному туннелю является гибридной. Она сочетает стандартную IP-маршрутизацию и технику виртуальных каналов с коммутацией по меткам. Действительно, инкапсулирующие пакеты продвигаются от начальной к конечной точке туннеля GTP с помощью стандартной маршрутизации, выполняемой промежуточными маршрутизаторами магистральной сети. В то же время шлюзы и базовая станция коммутируют трафик на основе локальных меток, которыми, по сути, являются идентификаторы TEID и C-RNTI.

Нужно подчеркнуть, что IP-пакеты пользователя всегда проходят через шлюз PGW. Он является **якорем маршрутизации**, называемым также IP-якорем, при любых передачах данных. Это означает, что даже в том случае, когда телефон A обменивается данными с телефоном B и оба телефона в этот момент времени присоединены к базовым станциям, обслуживаемым одним и тем же шлюзом SGW, их IP-пакеты проходят от этого шлюза SGW через туннель P-S к шлюзу PGW, далее маршрутизируются обратно через туннель к шлюзу SGW, а затем по туннелю S-B направляются к их базовой станции. Обязательное вовлечение шлюза PGW в передачу пользовательского трафика объясняется не только тем, что он маршрутизирует трафик (и он находится в нужном для этого месте, так как бóльшая часть пользовательского трафика направляется в интернет), но и тем, что он занимается контролем и учетом интернет-трафика пользователя — проверяет, какой тип и интенсивность трафика разрешены для данного пользователя и сколько пользователь должен за него платить (это он делает совместно с некоторыми другими элементами сети).

Телефон может иметь несколько составных туннелей, связывающих его с пакетным шлюзом PGW. Каждый составной туннель может использоваться для передачи трафика с разными требованиями QoS. Обычно телефон имеет два составных туннеля: один для передачи трафика данных, а второй — для передачи голосового трафика.

Эстафетная передача и роуминг

Эстафетная передача (ЭП) является еще одним основным механизмом поддержания мобильности в сотовых сетях. Эстафетная передача происходит в сети LTE каждый раз, когда активное устройство пользователя переходит из одной соты в другую.

При этом сеть делает все возможное, чтобы активная сессия пользователя (веб-сессия или голосовая) не прерывалась. Для этого сеть старается сделать время передачи минимальным и инструктирует устройство пользователя перестать передавать данные старой соте и начать их передавать новой только в том момент, когда в новой соте и в маги-

стральной сети уже выделены и сконфигурированы ресурсы для передачи трафика по новому маршруту.

Эстафетная передача в сети LTE происходит по *инициативе и под управлением сети*, при этом сеть принимает решение на основе информации о качестве сигналов, получаемых устройством пользователя от нескольких соседних сот.

Существует несколько сценариев эстафетной передачи. Они определяются тем, какие узлы сети остаются обслуживать трафик переместившегося из соты в соту устройства пользователя («старые» узлы), а какие узлы сети были привлечены к обслуживанию трафика этого устройства из-за его перемещения («новые узлы»). Кроме того, сценарии отличаются набором действий, которые должны выполнить «старые» и «новые» узлы по реконфигурации сети для обеспечения эстафетной передачи.

Сценарий ЭП1

Пусть телефон T1 перемещается из соты 1 исходной базовой станции 1 в соту 2 целевой базовой станции 2, при этом базовые станции соединены непосредственно интерфейсом (рис. 24.9). Обе базовые станции находятся под управлением того же узла мобильности MME и направляют трафик в интернет через одни и те же шлюзы SGW и PGW (переход ЭП1 на рис. 24.9).

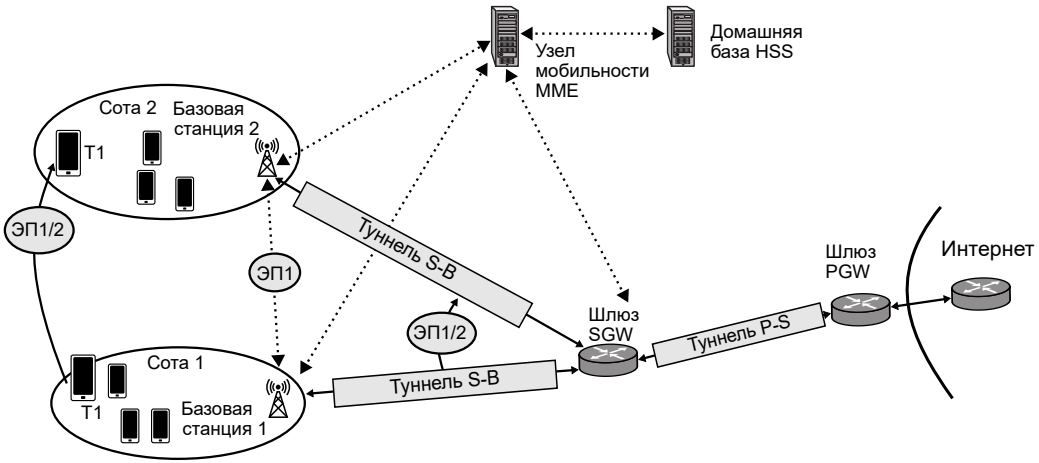


Рис. 24.9. Сценарии эстафетной передачи

Перед перемещением телефона в передаче его трафика принимали участие следующие узлы: базовая станция 1, шлюз SGW, шлюз PGW. Эти узлы находились под управлением сервера MME. При этом телефон и базовая станция 1 разделяли сессионные ключи шифрования и обеспечения целостности, с помощью которых выполнялась шифрация трафика в радиосреде (не путать с временным локальным идентификатором). Напомним, что эти ключи были выработаны в процессе аутентификации пользователя при присоединении телефона к сети.

После выполнения эстафетной передачи перестали участвовать в передаче трафика телефона T1 базовая станция 1 и туннель S-B 1. Кроме того, перестали использоваться сесси-

онные ключи, которые с телефоном разделяла базовая станция 1, в целях безопасности они вырабатываются заново новой базовой станцией в процессе эстафетной передачи.

Теперь в передаче трафика телефона T1 участвуют:

- Старые узлы: SGW, PGW, MME.
- Новые узлы и элементы: целевая базовая станция 2, туннель S-B 2, идентификатор C-RNTI2, сессионные ключи шифрования и обеспечения целостности данных.

Телефон T1 постоянно измеряет качество сигнала от всех сот в его зоне покрытия и посылает отчеты об измерениях той базовой станции, к которой он присоединен, в данном сценарии базовой станции 1.

В том случае, когда при перемещении телефона сигнал от базовой станции 2 становится лучше сигнала от базовой станции 1 на достаточно долгое время, базовая станция 1 принимает решение о совершении эстафетной передачи телефона T1 в соту 2 базовой станции 2.

После принятия такого решения исходная базовая станция 1 обращается к целевой базовой станции 2 через межстанционный интерфейс с запросом о передаче телефона T1 в соту 2. При этом целевой базовой станции сообщается, какие радиоресурсы потребуются телефону в соте 2.

Если целевая базовая станция 2 решает, что у соты 2 достаточно ресурсов для запрошенной передачи, то она посылает положительный ответ исходной базовой станции 1 и сообщает ей все параметры соты 2, необходимые телефону T1 для того, чтобы начать работать с сотой 2, включая новый временный идентификатор устройства C-RNTI2, который будет локальным адресом телефона в соте 2. После этого целевая базовая станция 2 заранее готовится к присоединению нового телефона, создавая для него необходимые структуры данных, включая новые сессионные ключи шифрования и обеспечения целостности. Они вычисляются независимо телефоном и целевой базовой станцией 2.

Далее исходная базовая станция 1 оповещает телефон T1 о том, что он может выполнять переход в соту 2, и передает устройству все параметры соты 2, полученные от целевой базовой станции 2. После этого телефон T1 немедленно прекращает передавать данные соте 1 и после синхронизации с базовой станцией 2 начинает передавать данные уже ей. Полная повторная процедура присоединения к сети, описанная выше, не выполняется.

Кроме добавления данных о телефоне T1 в информационные структуры базовой станции 2, для выполнения эстафетной передачи ЭП1 необходимо также создание нового туннеля S-B 2 для передачи трафика телефона T1 между целевой базовой станцией 2 и шлюзом SGW. Этот процесс инициирует целевая базовая станция 2, посылая запрос на создание туннеля серверу MME. Однако этот запрос целевая базовая станция делает только после того, как процесс перехода телефона T1 из соты 1 в соту 2 завершен. В течение времени перехода пакеты для телефона T1 от шлюза SGW продолжают поступать на исходную базовую станцию 1, и она переправляет их целевой базовой станции 2 по межстанционному интерфейсу, оставаясь на время якорем для трафика телефона в нисходящем направлении (очевидно, что в восходящем направлении телефону такое отклонение маршрута не требуется).

Получив запрос на создание нового туннеля, сервер MME оповещает шлюз SGW о том, что конечная точка туннеля S-B меняется — вместо базовой станции 1 ею теперь должна быть базовая станция 2.

Информация о локации телефона T1 обновляется только в сервере MME (теперь телефон T1 привязан к базовой станции 2). В домашней базе HSS обновление не требуется, так как телефон T1 не сменил сервер MME.

Процесс передачи обычно занимает 20–150 мс, а перерыв в доступе к телефону в радиосреде — 20–80 мс.

Сценарий ЭП2

Это тот же сценарий, что и ЭП1, за исключением того, что базовые станции не соединены прямым интерфейсом.

Теперь базовые станции общаются не непосредственно друг с другом, а через посредника — узел MME. При этом информация, которой обмениваются исходная и целевая базовые станции, остается той же.

На время перехода телефона между сотами узел MME создает временный непрямой туннель между исходной и целевой базовыми станциями для передачи телефону T1 нисходящего трафика. Этот туннель проходит через шлюз SGW, который играет роль промежуточного узла. После перестройки туннеля S-B (изменения его конечной точки на базовую станцию 2) временный туннель ликвидируется.

Мы не будем рассматривать более сложные сценарии эстафетной передачи, когда телефон переходит в соту, которая находится под управлением другого узла MME или другого шлюза PGW. Отметим только, что в этих сценариях эстафетная передача не всегда возможна, например, когда между узлами MME нет интерфейсной связи. Поэтому вместо эстафетной передачи в этом случае происходит полное повторное присоединение устройства пользователя к сети.

Общим недостатком описанных методов эстафетной передачи является обязательный период потери связи между устройством пользователя и сетью. Эта потеря обусловлена тем, что во всех сценариях эстафетная передача происходит по принципу «разрыв перед созданием связи» (break-before-make) — устройство сначала перестает синхронизироваться со старой сотой и только потом входит в синхронизацию с новой и начинает с ней работать. Конечно, во всех описанных методах сеть старается заранее сделать как можно больше подготовительных работ в новой соте и в магистральной сети, чтобы разрыв связи был по возможности очень непродолжительным. Но полностью устранить его невозможно, если не разрешить устройству пользователя поддерживать связь с двумя сотами одновременно. Такое свойство называется двойным подключением (dual connectivity), и оно обеспечивает действительно бесшовную эстафетную передачу, потому что устройство пользователя постоянно имеет связь с сетью и при разрыве первичной связи со старой сотой трафик продолжает передаваться по вторичной связи с новой сотой. Такое свойство было стандартизовано в поздних версиях стандартов сетей LTE, но на практике стало применяться только в сетях 5G, поэтому мы отложим его рассмотрение до следующего раздела.

Роуминг

Роуминг — это переход устройства пользователя в сеть другого оператора связи. Роуминг может происходить между операторами сетей одной и той же страны, а также быть международным. Хотя роуминг может казаться похожим на эстафетную передачу, но это

принципиально другая процедура: при роуминге задача сохранения текущей сессии не ставится. Роуминг — это новое полное присоединение устройства пользователя к сети со всеми этапами.

Когда устройство пользователя присоединяется к сети другого оператора, его узел MME находит по глобальному идентификатору пользователя IMSI домашнюю сеть пользователя и домашнюю базу в ней, которая и обеспечивает узел MME контекстом безопасности пользователя для проведения аутентификации и последующего производства сессионных ключей. Далее процесс присоединения протекает стандартным образом. Исключением является сохранение домашнего шлюза PGW как якоря интернет-трафика, то есть туннели создаются между шлюзом SGW гостевого оператора и шлюзом PGW домашнего оператора.

Спящий режим и пейджинг

Если устройство некоторое время не проявляет активности, то сеть решает перевести его в спящее состояние.

Базовая станция следит за активностью устройств пользователя, находящихся в ее соте, с помощью таймера активности устройства. Если у какого-то устройства этот таймер истекает (его значение устанавливается оператором сети, оно обычно составляет от 5 до 20 секунд), то базовая станция передает устройству инструкцию перейти в спящее состояние, называемое также состоянием простоя.

Базовая станция освобождает все ресурсы радиосреды, выделенные пользователю, в том числе его идентификатор C-RTNI, а сервисный шлюз SGW разрушает туннель S-B, обслуживавший трафик устройства пользователя. Туннель P-S между шлюзом SGW и PGW для устройства сохраняется, как сохраняется и его IP-адрес, — для быстрого переприсоединения устройства к сети, если ему потребуется стать снова активным (для него пришел извне IP-пакет или же оно само сгенерировало IP-пакет для передачи).

Перевод устройства из активного состояния в спящее преследует две цели:

- экономия батарей устройства пользователя;
- освобождение дефицитных ресурсов радиосреды для активных пользователей.

В спящем состоянии приемопередатчик устройства пользователя бóльшую часть времени отключен. Он включается в следующих случаях:

- Во-первых, когда необходимо передать базовой станции обновленную информацию о своем местоположении — телефон может сделать это даже в спящем состоянии. Однако в спящем состоянии телефон делает это реже, не при каждом переходе из соты в соту, а только тогда, когда переход значительный: из одной определенной оператором сети группы сот в другую. Такие группы сот называются **областями отслеживания** (Tracking Area, TA) — это группа сот, объединенных оператором по какому-то признаку, чаще всего по признаку географической близости. Такой прием, с одной стороны, экономит энергию батареи, а с другой — не засоряет сеть ненужными сообщениями.
- Во-вторых, устройство периодически включает свой радиоблок для того, чтобы проверить, не пришло ли для него **пейджинговое сообщение** (page message), то есть сообщение вызова, которое генерирует сервер MME. Генерация пейджингового сообще-

ния происходит, когда из интернета для спящего устройства на пакетный шлюз PGW поступает IP-пакет. Шлюз PGW, не зная, что устройство находится в спящем режиме, передает его по сохранившемуся туннелю P-S до сервисного шлюза SGW. Шлюз SGW не может передать его дальше (туннель к базовой станции для спящего телефона разрушен), поэтому он обращается к серверу MME за помощью, и тот генерирует и отсылает через базовую станцию пейджинговое сообщение устройству пользователя, говорящее о том, что устройству надо перейти в активный режим.

При отправке пейджингового сообщения сервер MME, ответственный за этот процесс, точно не знает, в какой соте находится в данный момент устройство пользователя. Он знает лишь, в какой области отслеживания оно находится. Поэтому по команде MME пейджинговое сообщение посылают все базовые станции, обслуживающие данную область отслеживания.

Получив пейджинговое сообщение, устройство посылает базовой станции запрос на восстановление ее радиоресурсов. Затем базовая станция восстанавливает эти ресурсы, а сервер MME восстанавливает туннель S-B к базовой станции — и устройство пользователя получает IP-пакет, из-за прибытия которого его разбудили.

Радиосеть доступа

Радиосеть доступа LTE — это одношаговая (single-hop) инфраструктурная сеть, в которой все устройства пользователя обмениваются данными только с базовой станцией — элементом инфраструктуры.

Частоты и мультиплексирование

Оператору сети LTE регулирующим органом страны может быть выделена полоса частот различной ширины — от 1,25 МГц до 20 МГц (в различных диапазонах частот, от 400 МГц до 3700 МГц), но наиболее часто выделяется полоса 10 МГц. Передача и прием данных происходят в различных частотных диапазонах, то есть оператору может быть выделена полоса шириной 10 МГц для передачи в диапазоне 1800–1810 МГц и полоса шириной 10 МГц для приема в диапазоне 1850–1860 МГц.

В нисходящем направлении радиосреда разделяется между устройствами пользователя с помощью техники мультиплексирования OFDMA. Ширина подканала OFDMA в сети LTE всегда равна 15 кГц, поэтому в зависимости от ширины диапазона для передачи пользовательских данных оператор может использовать от 76 (диапазон шириной 1,25 МГц) до 1200 подканалов (диапазон шириной 20 МГц). В соответствии с принципом работы OFDMA частота модуляции сигнала в подканале должна быть равна ширине частотной полосы подканала, то есть 15 кГц.

Скорость передачи данных в одном подканале определяется количеством состояний сигнала примененного метода модуляции, например, для кода QPSK с двумя состояниями сигнала скорость будет равна 15 (частота модуляции) × 2 = 30 Кбит/с, для кода 16-QAM (4 состояния) — 15 × 4 = 60 Кбит/с, а для кода 64-QAM (6 состояний) — 15 × 6 = 90 Кбит/с.

Каждый частотный подканал радиосети LTE разделяется в технике временно́го мультиплексирования TDM на тайм-слоты длительностью 0,5 мс, в одном тайм-слоте переносится 7 символов кода данных. Небольшая длительность тайм-слота обеспечивает

быстрое восстановление искаженных помехами данных. Восстановление происходит на уровне так называемых подкадров, в которые входит два тайм-слота, образующих подкадр длительностью 1 мс. Десять подкадров объединяются в кадр. Начало кадра является синхронизирующим моментом для нахождения определенного подкадра -- это необходимо в некоторых процедурах радиосети.

Единицей выделения пропускной способности радиосреды LTE устройству пользователя является ресурсный блок (Resource Block, RB) (рис. 24.10), состоящий из 12 подканалов и одного тайм-слота (7 символов). Двенадцать подканалов образуют канал шириной 180 кГц. В ресурсном блоке помещается 84 символа, а количество бит в ресурсном блоке зависит от типа модуляции. При модуляции 64-QAM (6 бит на символ) их там помещается 504.

В сообщениях протоколов радиосреды ресурсные блоки идентифицируются номерами, от 0 до N, при ширине полосы частот в 10 МГц N=50 (часть боковых частот оператором не используется, они являются защитными от сигналов других операторов — соседей по спектру).

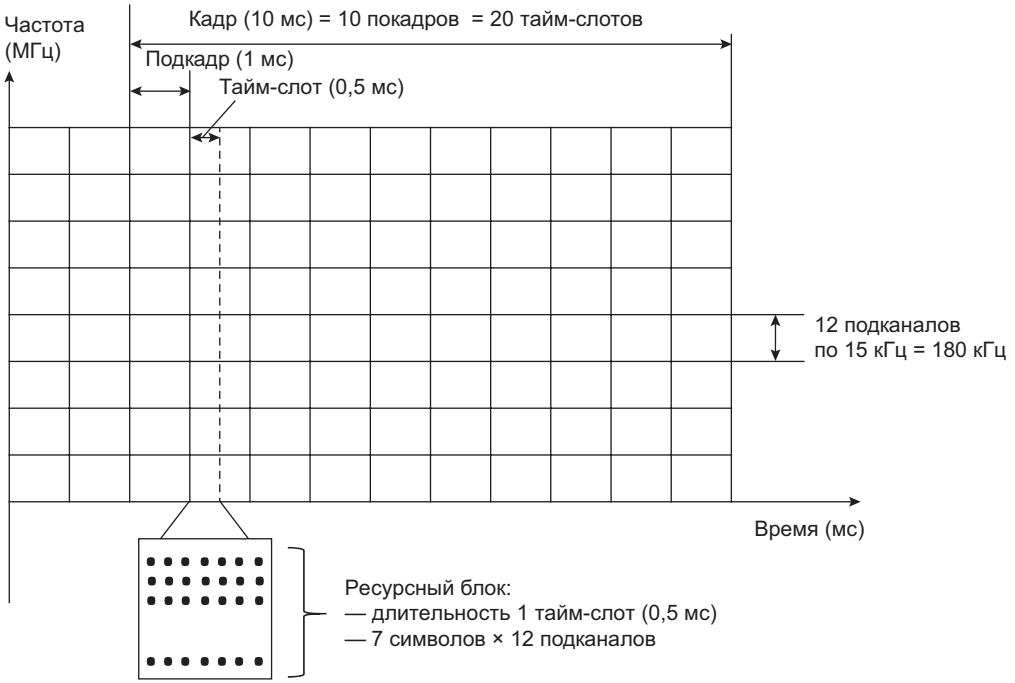


Рис. 24.10. Ресурсный блок радиосети LTE

Нетрудно подсчитать, что максимальная скорость передачи данных от сети к устройству пользователя при постоянном выделении ему одного ресурсного блока, использующего модуляцию 64-QAM (1 символ — 6 бит), составляет:

$$6 \times 84 \times 2000 = 1008 \text{ Кбит/с} \approx 1 \text{ Мбит/с}.$$

Каждому пользователю может быть выделено определенное количество ресурсных блоков, номера которых базовая станция сообщает устройству пользователя по каналу управления, как это было показано выше при описании процедуры присоединения устройства пользователя к соте. Распределение ресурсных блоков между пользователями происходит динамически и постоянно: каждую миллисекунду базовая станция принимает решение о перераспределении ресурсов радиосреды между пользователями соты и сообщает о своем решении устройствам пользователей.

Если все ресурсные блоки радиосреды выделяются одному устройству пользователя, то при ширине частотного диапазона провайдера 20 МГц в распоряжении базовой станции имеется 100 (1200 : 12) ресурсных блоков, и максимальная скорость передачи данных (при идеальных условиях радиоэфира) составит 100 Мбит/с.

В центре диапазона частот для нисходящей передачи выделяется полоса шириной в шесть ресурсных блоков для передачи синхросигналов соты и общей информации о соте. Кроме этого широковещательного канала в нисходящем направлении других специальных выделенных частотных каналов не существует, и все типы данных передаются в частотах, предназначенных для ресурсных блоков данных.

В восходящем направлении радиосреда разделяется на ресурсные блоки с помощью другой техники мультиплексирования — техники SC-FDMA (Single Carrier FDMA). Основным преимуществом техники SC-FDMA по сравнению с OFDMA является ее более низкое отношение пиковой мощности передатчика к его средней мощности. По этой причине передатчик SC-FDMA потребляет меньше энергии, и это достоинство оказалось решающим при выборе технологии мультиплексирования для передатчиков устройств пользователя в технологии LTE. Для передатчика базовой станции, питающегося от сети, это свойство преимуществом не является. Поэтому для него была выбрана техника OFDMA, которая позволяет более гибко отображать потоки символов отдельного подканала на антенны MIMO.

В частотном диапазоне, выделенном для восходящей передачи, существует два специальных частотных канала, отделенных от основной полосы частот, — канал управляющих данных и канал случайного доступа к соте. В первом канале устройства пользователя передают базовой станции служебные кадры ACK/NACK, сообщающие об успешности приема блоков данных. Второй канал предназначен для передачи устройством пользователя преамбулы, говорящей базовой станции, что устройство хочет присоединиться к соте.

Стек протоколов сети радиодоступа

Радиосреда LTE с ее делением на ресурсные блоки и их динамическим выделением пользовательскими устройствами с помощью техник мультиплексирования OFDMA и SC-FDMA является весьма сложной средой, что определяет сложную организацию стека ее протоколов (рис. 24.11). Начнем рассмотрение протоколов «сверху вниз».

Протокол NAS (Non-Access Stratum) является протоколом слоя управления. Он работает только в стеке устройства пользователя и является в этом стеке протоколом самого высокого уровня. Протокол NAS поддерживает функции управления мобильностью, взаимодействуя непосредственно с сервером мобильности MME (регистрация, аутентификация и эстафетная передача).

Протокол согласования с пакетными данными PDCP (Packet Data Convergence Protocol) обеспечивает подготовку IP-пакетов пользовательского слоя для передачи их по радиосреде, компрессируя заголовки протоколов IP/TCP/RTP. Этот протокол также обеспечивает правильную (упорядоченную) последовательность передачи пакетов на верхний уровень за счет их нумерации (но не восстанавливает потерянные пакеты).

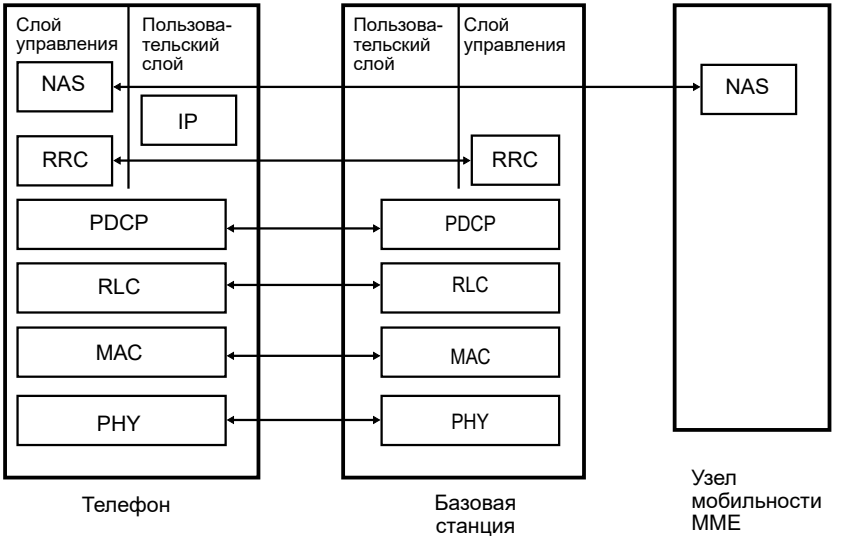


Рис. 24.11. Протоколы радиосреды LTE

Протокол RRC (Radio Resource Control) занимается управлением ресурсами радиосреды и работает как на базовой станции, так и на устройстве пользователя. С помощью этого протокола устройство пользователя общается с базовой станцией, запрашивая и получая в свое распоряжение сегмент виртуального носителя радиосреды (радио-ВН) и выполняя эстафетную передачу (ее часть, относящуюся к радиосреде).

Протокол управления радиосвязью RLC (Radio Link Control) выполняет функции по сегментированию сообщений, полученных от уровня PDCP, на более мелкие единицы, блоки RLC, а также по надежной передаче этих блоков через радиосреду. Можно сравнить эти функции с функциями транспортного протокола TCP, но TCP делает больше — он кроме надежной передачи еще и управляет скоростью передачи данных в сеть. Протокол RLC выбирает размер своих блоков таким, чтобы они могли поместиться в ресурсные блоки RB радиосреды. Надежную передачу блоков RLC обеспечивает механизм автоматического запроса повторной передачи ARQ (Automatic Retransmission reQuest). Этот механизм основан на квитировании последовательно пронумерованных передаваемых блоков PDU. Для управления потоком блоков используется механизм скользящего окна.

Протокол управления доступом к среде MAC работает как в слое управления, так и в пользовательском слое. Основной функцией этого протокола является распределение ресурсных блоков радиосреды RB между данными различных пользователей. Эта функция слоя управления реализована в MAC-планировщике базовой станции, так как только базовая станция принимает решения о том, какие блоки данных, поступающие от уровня RLC, нуж-

но поместить в блоки OFDMA в данном цикле планирования. Планирование осуществляется каждую миллисекунду, то есть для каждого подкадра, — это довольно трудная работа для базовой станции. Но такая тонкая дискретность во времени позволяет удовлетворять требования к качеству обслуживания трафика разных классов. При принятии решения о планировании MAC-планировщик принимает во внимание различные факторы, такие как приоритет потока данных (из QCI виртуального канала), данные о качестве радиоканала между базовой станцией и устройством пользователя CQI, периодически получаемые от устройства пользователя, а также о заполненности буферов данных устройства пользователя и передатчика базовой станции.

Уровень MAC выполняет также функцию быстрой коррекции данных, используя методы коррекции FEC. Отправив блок данных физическому уровню, протокол MAC организует его надежную доставку с помощью гибридной процедуры **HARQ** (Hybrid ARQ). Процедура названа гибридной, потому что она совмещает два подхода к восстановлению потерянных или искаженных кадров — технику квитанций и технику кодов FEC. Перед отправлением блока данных для него вырабатывается код FEC, который также помещается в блок. При приеме блока данных MAC-уровень проверяет его корректность по коду FEC и в том случае, если данные в блоке не искажены или же поддаются исправлению с помощью кода FEC, отправляет положительную квитанцию ACK. В противном случае отправляется отрицательная квитанция NACK. Квитанции получатель отправляет через 4 блока после отправления исходного. Эта пауза дается получателю для обработки блока — вычислению FEC и попытке исправления искаженных бит. Особенностью процедуры HARQ является то, что после отправления отрицательной квитанции получатель не отбрасывает блок, как это происходит во многих других реализациях идеи применения квитирования, вместо этого блок сохраняется в буфере. Отправитель, получив отрицательную квитанцию, снова передает тот же блок, при этом он может вычислить значение FEC другим способом, чтобы увеличить вероятность успешного исправления кода блока получателем. Если получатель смог исправить искаженные биты при повторном получении блока, то он отправляет положительную квитанцию. Если же процесс ретрансляции проблемного блока повторяется несколько раз, то при достижении максимального числа попыток (оно устанавливается базовой станцией) блок отбрасывается.

Процессы HARQ на уровне MAC и ARQ на уровне RLC дополняют друг друга. Восстановление блоков с помощью HARQ быстрее, чем с помощью ARQ, так как HARQ работает на более низком уровне, где блоки передаются с периодом в 1 мс. Кроме того, HARQ применяет FEC, и некоторые искаженные блоки удается восстановить без повторной передачи. ARQ работает на более медленном, верхнем уровне и исправляет ситуацию в том случае, когда быстрому HARQ не удалось ее исправить.

Физический уровень (PHY) занимается модуляцией (QPSK, 16-QAM, 64-QAM или 256-QAM), кодированием и передачей радиосигналов с использованием технологий OFDMA и MIMO.

Передача голоса через сеть LTE

Все, что рассматривалось в этой главе ранее, относилось к тому, как сеть LTE обеспечивает доступ мобильному устройству к интернету. И ничего не говорилось о том, что самым непосредственным образом связано с сотовыми сетями, а именно о телефонных разговорах. В этом разделе мы изучим именно этот аспект работы мобильной сотовой сети LTE.

Сеть LTE для установления *телефонных сессий* между устройствами ее пользователей, а также мультимедийных сессий другой природы, например видеосвязи, использует отдельную **мультимедийную IP-систему** (IP Multimedia System, **IMS**). Основным результатом работы системы IMS является сохранение традиционной, привычной и дружественной для пользователя процедуры телефонного разговора. Это означает, что пользователь обычным образом набирает на своем телефоне номер вызываемого абонента и далее участвует в знакомой процедуре установления соединения по схеме «запрос-ответ», когда телефон вызываемого абонента оповещает своего пользователя звонком или вибрацией о приходе запроса на установление сеанса связи, а в это же время телефон вызывающего абонента длинными гудками сигнализирует, что надо ждать ответа, или же короткими гудками оповещает о занятости абонента.

При внешней неизменности традиционной процедуры телефонного разговора система IMS реализует голосовую связь абонентов совсем иными средствами, отличными от средств традиционной телефонии. Для решения поставленной задачи система IMS использует пакетную сеть LTE. Как мы знаем, сеть LTE обеспечивает взаимодействие пользователей на основе IP-адресов их устройств, поэтому при обращении к услугам этой сети система IMS выполняет преобразование телефонных номеров абонентов в соответствующие IP-адреса. После того как IP-адреса определены, сеть LTE обеспечивает передачу голосового трафика между устройствами с этими адресами, поддерживая мобильность и требуемое качество обслуживания, то есть выполняет свою обычную работу. Нужно подчеркнуть, что услугу телефонных разговоров VoLTE (голос через сеть LTE) система IMS и сеть LTE организуют *совместно*, то есть справедлива формула VoLTE = LTE + IMS.

В системе IMS для установления мультимедийных соединений применяются протоколы IP-телефонии, разработанные еще в конце 90-х. Основным служебным протоколом IP-телефонии является протокол установления мультимедийных сессий SIP, заменяющий традиционные сигнальные протоколы телефонных сетей (о них вы можете прочитать на сайте книги).

(S) *Сети ISDN*

Протокол SIP

Протокол SIP (Session Initiation Protocol) — это *сигнальный протокол*, то есть протокол, который отвечает за установку, управление и завершение мультимедийных сессий в IP-сетях, но в его функции не входит передача, собственно, медиа данных (голоса или видео). Этот протокол разработан в расчете на определенную инфраструктуру, в которую входят функционально различные серверы. Поэтому можно говорить и о SIP-системе, имея в виду комплекс функциональных узлов, работающих на основе протокола SIP.

Рассмотрим основные идеи и механизмы этого протокола и соответствующей SIP-системы на примере ранней версии этого протокола, которая была стандартизирована IETF в RFC 3261. Затем в следующем разделе мы рассмотрим версию SIP для системы IMS, которая не переопределяет SIP RFC 3261, но дополняет и профилирует его для лучшей интеграции с мобильными сотовыми сетями.

В функции протокола SIP входит:

- Установление сессии связи — локализация и вызов абонента, согласование параметров связи, например алгоритма кодирования и декодирования голоса.

- Управление сессией — удержание и перевод вызова, добавление видео, приостановка и возобновление сессии.
- Штатное завершение сессии — после окончания разговора сессия разрывается путем передачи соответствующего SIP-сообщения.
- Регистрация пользователей и определение их местоположения

После того как протокол SIP установил соединение, абоненты обмениваются голосовыми или видеоданными с помощью других протоколов. Обычно это делается с помощью протокола RTP (Real-time Transport Protocol), определенного в RFC 3550.

Ниже приводятся типичные сообщения, которые протокол SIP использует для выполнения своих функций:

- REGISTER — регистрация устройства;
- INVITE — запрос на установление сеанса связи, передаваемый от вызвавшего телефона вызываемому;
- TRYING — вызываемый телефон получил сообщение INVITE;
- RINGING — вызываемый телефон «звонит», то есть передает сигнал вызова своему абоненту;
- OK — звонок принят вызываемым абонентом;
- ACK — сессия установлена, можно передавать данные;
- BUSY HERE — вызываемый телефон занят;
- CANCEL — остановка вызова до получения ответа;
- BYE — завершение сеанса.

Адресами абонентов в протоколе SIP являются универсальные идентификаторы (URL), такие как sip:bill@ja.net или sip:bob@mgu.ru. Напомним, что URL-имена используются и в других интернет-службах, например в электронной почте или протоколе HTTP. URL-имя состоит из нескольких частей: имени протокола, который нужно использовать для доступа к объекту (в приведенных выше именах это sip), пользовательской части имени (bill и bob) и имени соответствующего домена DNS, здесь ja.net и mgu.ru. Применение URL-имен позволяет использовать систему DNS интернета для поиска IP-адресов узлов системы SIP.

Частным случаем URL-имени является телефонный номер, который представляет собой пользовательскую часть URL-имени, например tel:+1234567890, а остальные части отсутствуют. Отсутствие DNS-части затрудняет использование системы DNS для отображения имени в форме телефонного номера на IP-адрес. Мы рассмотрим, как решается эта проблема, позже. А сейчас в приведенном ниже описании работы протокола SIP будут использоваться только URL-имена с DNS-частью. Первую часть этих имен, то есть имя sip, мы для краткости опустим, так как все имена в примере имеют этот префикс.

Основными элементами протокола SIP являются:

- **Агент пользователя** (User Agent, UA) — специальное программное обеспечение, реализующее протокол SIP. Устанавливается на конечных узлах — телефон, программный телефон, приложение.
- **Сервер- регистратор SIP** — регистрирует местоположение пользователей.
- **Прокси-сервер SIP** — маршрутизирует SIP-сообщения.

На рис. 24.12 показана схема работы протокола SIP. Она включает в себя два конечных устройства пользователей, bill@ja.net и bob@piter.ru, устанавливающих сеанс голосовой связи. Эти устройства являются узлами обычной IP-сети, промежуточные маршрутизаторы между ними не показаны для сохранения ясности рисунка.

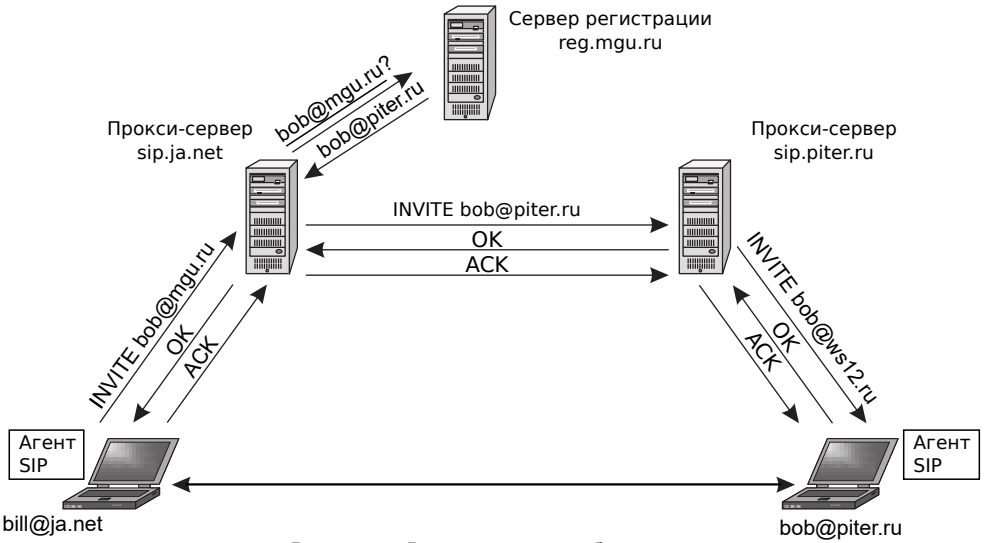


Рис. 24.12. Взаимодействие абонентов SIP

На каждом абонентском устройстве установлен агент пользователя SIP. Агенты SIP конечных устройств обычно не общаются друг с другом непосредственно[1]. Они обмениваются сообщениями протокола SIP через прокси-серверы.

Прокси-сервер SIP является для агента SIP пользователя посредником и помощником в общении с другими узлами системы SIP. Все сообщения агент SIP передает через назначенный ему прокси-сервер. Прокси-сервер выполняет действия, необходимые для маршрутизации запроса INVITE к агенту SIP вызываемого абонента. Конечное устройство может узнать об имени и IP-адресе назначенного ему прокси-сервера различными способами: от сервера DHCP (наряду с адресом маршрутизатора по умолчанию), используя запрос DNS, или же эта информация может быть сконфигурирована администратором сети. Способ получения информации о прокси-сервере в стандарте SIP IETF не оговаривается.

Прокси-сервер SIP управляет передачей трафика сигнальных сообщений в VoIP-сетях.

- Когда пользователь звонит, его SIP-агент отправляет запрос на прокси-сервер.
- Прокси-сервер анализирует запрос, определяет через серверы регистрации местоположение адресата, модифицирует сообщение и пересылает его дальше другому прокси-серверу.
- Ответные сообщения проходят обратный путь через те же прокси-серверы.

[1] Прямое взаимодействие конечных узлов технически возможно, но оно обычно блокируется из соображений безопасности.

- Мультимедийный поток данных передается напрямую между устройствами, минуя прокси-серверы

Серверы регистрации SIP хранят информацию о текущей точке присоединения конечного устройства к системе SIP, при этом под точкой присоединения понимается прокси-сервер, через который конечное устройство в данный момент общается с системой SIP. Подразумевается, что конечное устройство может изменить свою точку присоединения к системе SIP и начать общаться с узлами системы SIP через другой прокси-сервер. Именно эти изменения точки присоединения и отслеживает сервер регистрации. Информацию о точке присоединения сервер регистрации получает в ходе процедуры регистрации, которую инициирует агент SIP устройства пользователя. При активизации устройства (например, при включении питания) агент посылает сообщение REGISTER, в котором он указывает свое URL-имя. Это сообщение, как и все другие сообщения конечного устройства, поступают на прокси-сервер, который передает его *домашнему* серверу регистрации данного конечного устройства. В системе SIP может быть несколько серверов регистрации. У *домашнего* сервера регистрации доменная часть имени совпадает с доменной частью имени пользователя устройства. Например, у пользователя bob@mgu.ru доменной частью имени является имя mgu.ru, поэтому его домашним сервером регистрации является сервер регистрации с именем reg.mgu.ru. Сервер регистрации не интересуется IP-адресом конечного устройства. Он хранит только соответствие имени конечного устройства (точнее, URL-имени владельца этого устройства) URL-имени прокси-сервера, который обслуживает это устройство. Знание IP-адреса — это сфера ответственности прокси-сервера, обслуживающего конечное устройство.

Рассмотрим этапы работы протокола SIP при поступлении сообщения INVITE (см. рис. 24.12).

1. Абонент bill@ja.net хочет установить мультимедийный сеанс связи с абонентом bob@mgu.ru.
2. К этому моменту времени пользователь bob@mgu.ru прошел регистрацию с помощью сообщения REGISTER bob@mgu.ru в своем домашнем сервере регистрации reg.mgu.ru.
   - Для регистрации он использовал прокси-сервер sip.piter.ru, который был ему назначен, когда он подключил свой лэптоп к сети (очевидно, это была для него гостевая сеть).
   - Прокси-сервер sip.piter.ru использовал систему DNS для того, чтобы найти IP-адрес домашнего сервера регистрации пользователя reg.mgu.ru.
   - После его нахождения прокси-сервер sip.piter.ru передал запрос REGISTER bob@mgu.ru серверу регистрации reg.mgu.ru, тот успешно аутентифицировал пользователя bob@mgu.ru и запомнил, что он достижим через прокси-сервер sip.piter.ru.
3. Для установления сессии связи пользователь bill@ja.net направляет своему прокси-серверу sip.ja.net запрос INVITE bob@mgu.ru.
4. Прокси-сервер sip.ja.net анализирует доменную часть имени вызываемого абонента и с помощью системы DNS находит, что домашним для этого пользователя является сервер регистрации reg.mgu.ru.
5. Прокси-сервер обращается к серверу регистрации reg.mgu.ru с запросом о текущей точке присоединения устройства пользователя bob@mgu.ru, который возвращает ему ответ о том, что абонент bob@mgu.ru в данный момент зарегистрирован как активный

в домене piter.ru с именем bob@piter.ru. Протокол запросов о локации абонента между прокси-сервером и сервером регистрации в RFC 3261 не определен, и его реализация оставлена на усмотрение разработчиков программного обеспечения SIP. В варианте SIP IMS этот пробел устранен.

6. Прокси-сервер использует эту информацию для того, чтобы принять решение о дальнейшей маршрутизации сообщения INVITE. Для этого он использует новую доменную часть имени вызываемого абонента piter.ru и с помощью системы DNS находит, что этот домен обслуживает прокси-сервер с именем sip.piter.ru. Поэтому он направляет запрос этому прокси-серверу, указав в нем новое имя абонента: INVITE bob@piter.ru.
7. Прокси-сервер sip.piter.ru знает IP-адрес SIP-агента bob@piter.ru, так как он регистрировался через этот прокси-сервер, поэтому он предает сообщение INVITE bob@piter.ru непосредственно лэптопу, на котором работает агент SIP этого пользователя.
8. Далее протокол SIP работает, подобно большинству протоколов сигнализации: если пользователь bob@piter.ru соглашается принять вызов и снимает трубку своего SIP-телефона (или щелкает по соответствующему значку своего программного SIP-телефона), то он тем самым посылает ответ OK назад по цепочке прокси-серверов, через которые прошел запрос. В нашем случае это прокси-серверы sip.ja.net и sip.piter.ru. Чтобы сделать возможной реверсивную маршрутизацию, они при прямой передаче запроса INVITE добавляли в его заголовок свои имена. В ответе OK содержится IP-адрес агента SIP на лэптопе пользователя bob@piter.ru, так что после его получения агент SIP пользователя bill@ja.net может начать передачу голосового трафика, используя этот IP-адрес. Окончательное установление сессии фиксируется отправкой сообщения ACK (подтверждение) от вызывающего абонента к вызываемому.

После установления сессии обмен голосовыми данными происходит непосредственно между устройствами пользователей на основе IP-адресов этих устройств. Узлы системы SIP в этом обмене не участвуют. Оцифрованные голосовые данные передаются через сеть протоколом RTP, который переносит отметки времени и последовательные номера пакетов, помогая конечным узлам сеанса восстанавливать аналоговую информацию реального времени.

Мультимедийная система IMS

При разработке мультимедийной системы IMS за основу был взят рассмотренный выше протокол SIP, стандартизованный IETF. Но для обеспечения масштабируемости системы, требуемой для обслуживания тысяч одновременных соединений, к прокси-серверам и серверам регистрации были добавлены новые типы инфраструктурных элементов, а сам протокол SIP был модифицирован.

Главным отличием версии SIP IMS от SIP IETF является узкая специализация этого протокола. В то время как протокол SIP IETF позиционируется как гибкий стандарт для построения широкого спектра систем IP-телефонии в сетях самой различной природы и масштаба (корпоративных и домашних, проводных и беспроводных, мобильных и фиксированных), протокол SIP IMS предназначен исключительно для работы в тесной интеграции с сетью оператора мобильной связи. Такая специфика позволила исключить все опциональные возможности и неопределенности протокола SIP IETF и вместо этого построить жестко регламентированный, детерминированный протокол, который гаран-

тированно выполняет свои функции при предоставлении мультимедийных сервисов оператором связи.

Компоненты системы IMS

К числу основных компонентов IMS относятся серверы, выполняющие функции управления сессиями вызовов (Call Session Control Functions, **CSCF**). Существует три типа таких серверов:

- **P-CSCF (Proxy-CSCF)** — прокси-сервер системы IMS.
- **S-CSCF (Serving-CSCF)** — сервер регистрации и локализации пользователей.
- **I-CSCF (Interrogating-CSCF)** — запрашивающий сервер, занимается распределением серверов регистрации между пользователями.

Кроме того, в инфраструктуру входят:

- **База данных абонентов** (Home Subscriber Server, **HSS**) — общая с системой LTE база данных пользователей.
- **Серверы приложений** (Application Servers, **AS**) — предоставляют дополнительные услуги: голосовой почты, переадресации вызовов, конференц-связи, передачи текстовых сообщений.

Имена узлов в стандарте IMS довольно трудны для запоминания. Они имеют общую часть — CSCF, которая, как было сказано выше, говорит о том, что эти узлы выполняют некоторую часть общей функции управления сессией вызова. О том, какую часть этой функции они выполняют, говорит префикс имени — P (proxy), S (serving) или I (Interrogating).

Сравним функции узлов системы SIP IMS с функциями узлов аналогичного назначения системы SIP IETF.

P-CSCF (Proxy-CSCF) выполняет роль *прокси-сервера SIP*. Все запросы устройства пользователя на установление соединения с некоторым другим устройством, а также запрос на регистрацию должны проходить через сервер P-CSCF. Прямые обмены сигнальной информацией между конечными устройствами, возможные в системе SIP, здесь строго не допускаются, потому что общение через узлы-посредники позволяет контролировать SIP-соединения в соответствии с условиями подписки абонента на сервисы IMS, а при прямом взаимодействии абонентов это затруднительно.

Функции *сервера регистрации SIP* выполняют три сервера системы IMS разного типа. Основные действия по регистрации выполняет **сервисный сервер S-CSCF** (Serving-SCSF). Этот сервер принимает запросы на регистрацию и совместно с **домашней базой HSS** (второй сервер, участвующий в регистрации) выполняет аутентификацию пользователя услуг IMS. Эта аутентификация проводится независимо от аутентификации пользователя сети LTE, так как услуги, в целях которых проводится каждая аутентификация, различны.

Регистрация пользователя услуг системы IMS (SIP-регистрация) проводится сразу же после успешного присоединения телефона к сети LTE (описанного выше в этой главе). SIP-регистрация — это отдельная процедура, независимая от LTE-присоединения, со своим этапом аутентификации на основе имени пользователя IMS. SIP-регистрация проводится в том случае, если телефон в своем запросе на присоединение к сети LTE указал не только соединение с интернетом, но и соединение с ISM, что на практике всегда и происходит,

так как пользователи обычно подписываются на обе услуги — доступа к интернету и телефонные услуги.

Наличие сервера третьего типа, так называемого **запрашивающего сервера I-CSCF** (Interrogating-CSCF), объясняется тем, что в крупном домене IMS обычно применяется несколько серверов S-CSCF для баланса их загрузки, так как поток сообщений INVITE на установление соединений между телефонами может быть очень интенсивным и один сервисный сервер может с ним не справиться. Запрашивающий сервер участвует в процессе регистрации пользователя и совместно с базой данных HSS принимает решение, какой сервисный сервер S-CSCF нужно выделить этому пользователю для регистрации.

Упрощенно процедура регистрации пользователя в системе IMS может быть представлена в виде следующей последовательности шагов:

- SIP-агент пользователя подключается к назначенному ему прокси-серверу P-CSCF.
- SIP-агент отправляет прокси-серверу сообщение REGISTER user@domain.com.
- Прокси-сервер P-CSCF обращается с этим сообщением к запрашивающему серверу I-CSCF.
- Сервер I-CSCF обращается к базе данных HSS и получает от нее список серверов S-CSCF данного домена. База данных HSS вместе с именами серверов S-CSCF передает список предоставляемых этими серверами дополнительных функций по обработке соединений (перенаправление звонка, автоответчик и т. п.) и поддерживаемые типы аутентификации. Сервер I-CSCF на основе этих характеристик, а также информации о загруженности серверов S-CSCF назначает пользователю user@domain.com один из них для регистрации.
- Назначенный сервер S-CSCF совместно с базой данных проводит аутентификацию пользователя user@domain.com.
- В случае успешной аутентификации сервер S-CSCF отправляет прокси-серверу P-CSCF, который обращался к запрашивающему серверу, сообщение 200 OK как свидетельство успешной регистрации. При этом сервер S-CSCF запоминает тот факт, что пользователь user@domain.com зарегистрирован, а также запоминает имя прокси-сервера, которому надо передавать сообщения, адресованные пользователю user@domain.com.
- Прокси-сервер P-CSCF передает сообщение 200 OK SIP-агенту пользователя user@domain.com, и на этом процесс регистрации заканчивается.

В дальнейшем мы будем называть сервисный сервер сервером регистрации домена, учитывая его основную роль в этом процессе.

В том случае, когда прокси-серверу необходимо обслужить запрос INVITE user@domain.com, он обращается к запрашивающему серверу I-CSCF с просьбой указать, к какому конкретно серверу регистрации ему нужно обратиться, чтобы получить маршрут, по которому можно передать сообщения INVITE этому пользователю. Напомним, что маршрут в данном случае означает имя прокси-сервера, обслуживающего пользователя user@domain.com. После получения ответа от запрашивающего сервера взаимодействие между прокси-сервером и сервером регистрации идет уже непосредственно.

Система IMS обычно включает несколько серверов приложений (на рисунке не показаны), с помощью которых организуются дополнительные телефонные услуги: автоответчик, голосовая почта, переадресация вызовов и т. п. Кроме того, обычно в системе IMS имеются шлюзы для взаимодействия с телефонными сетями других типов.

В целом, несмотря на некоторые усложнения, вызванные необходимостью иметь хорошо масштабируемую систему, мультимедийная система IMS работает аналогично описанной выше более простой системе SIP.

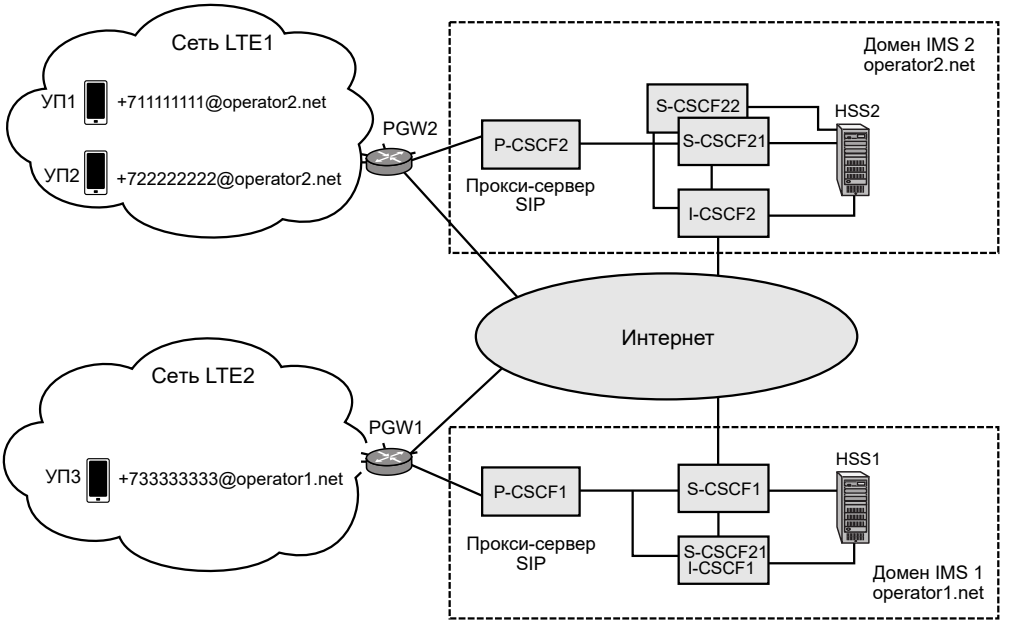


Рис. 24.13. Два домена системы IMS

На рис. 24.13 показаны две системы IMS, принадлежащие двум операторам сетей LTE, — система IMS 1 и система IMS 2. Как видно из рисунка, системы IMS достаточно автономны. Они имеют связи со своими сетями LTE только через шлюзы PGW, как и любые внешние по отношению к сети LTE узлы и сети. Через шлюзы PGW узлам системы IMS передаются сигнальные сообщения SIP, а после установления SIP-сессии пользовательские данные этой сессии идут через шлюзы PGW и интернет, минуя системы IMS. Внутреннее взаимодействие между узлами одной системы IMS происходит с использованием протоколов стека TCP/IP. Эти узлы могут располагаться как в пределах внутренней IP-сети оператора (это более надежный вариант, гарантирующий низкие задержки и потери пакетов), так и в различных центрах данных, например в облаке, и в этом случае они используют менее скоростные и надежные соединения публичного интернета.

Доменные имена

В системе IMS **доменные имена** играют важную роль, так как их корректное применение позволяет узлам систем IMS различных операторов пользоваться услугами системы DNS для выполнения различных операций: роуминга, маршрутизации SIP-сообщений, аутентификации и др.

Понятие **домена** в системе IMS совпадает с понятием домена в системе DNS. Домен IMS объединяет все узлы системы IMS и дает им общее доменное имя верхнего уровня. На-

пример, в системе IMS оператора 1 у всех узлов доменное имя верхнего уровня одно и то же — это operator1.net. Поэтому имена узлов этой сети имеют вид p-cscf1.operator1.net, s-cscf1.operator1.net и т. д. Эта система имен узлов позволяет использовать запросы DNS для нахождения IP-адресов функциональных узлов определенного типа, таких как прокси-серверы и серверы регистрации, в сети определенного оператора.

Доменные имена пользователей в системе IMS бывают двух типов: *публичные* и *частные*. Оба имени хранятся в SIM-карте телефона и в домашней базе оператора HSS. Публичное имя состоит из индивидуального имени пользователя (например, +711111111) и имени домена его домашнего оператора (например, operator2.net). Публичное имя используется во всех сообщениях SIP в открытом виде — то есть в виде текста. Благодаря ему сообщения SIP легко понимать. Частное имя в открытом виде никогда не передается и используется только в процессе аутентификации пользователя при его регистрации в системе IMS.

Правило использования доменных имен в системе IMS формулируется следующим образом:

- Пользователь может иметь несколько различных публичных URL-имен, однако в системе IMS все они должны иметь одно и то же доменное имя, а именно доменное имя оператора системы IMS.
- Все серверы системы IMS также должны использовать доменное имя оператора системы IMS в качестве старшей части своего доменного имени.
- Менять доменное имя пользователя, как это разрешено в системе SIP IETF (и как показано на рис. 24.12) в системе IMS нельзя, так как весь механизм нахождения конечного устройства по публичному имени его пользователя построен на неизменности его доменной части.

Роль DNS в работе системы IMS

Система IMS пользуется услугами системы DNS для решения двух задач:

- Отображения телефонного номера абонента на его URL-имя.
- Поиск IMS-сервера определенного типа (запрашивающего сервера, прокси-сервера, сервера регистрации) в системе IMS определенного оператора.

Первая задача естественным образом вытекает из стремления предоставить пользователю привычный интерфейс работы с телефоном, включающий набор традиционного телефонного номера вместо URL-имени. Так как основные механизмы системы IMS базируются на URL-именах (без них маршрутизация сообщений и локализация пользователя невозможны), то необходимо найти способ отображения телефонного номера на URL-имя. Возвращаясь к примеру сети на рис. 24.13, эта задача состоит в том, чтобы разрешить пользователю +733333333@operator1.net инициировать соединение с пользователем +711111111@operator2.net, просто набирая на своем телефоне номер +711111111, а затем выполнить отображение +711111111 → +711111111@operator2.net и далее использовать во внутренних операциях это URL-имя пользователя.

Существует два способа отображения телефонного номера абонента на доменное имя типа +711111111@operator2.net. Первый основан на функции ENUM системы DNS интернета. Эта функция использует обратную зону DNS, которая отображает телефонные номера на доменные имена обладателей этих номеров. Однако операторы сотовых мобильных сетей

предпочитают обходиться без использования техники ENUM из-за ее плохой масштабируемости (нужно хранить таблицы отображения для всех имеющихся номеров, которых может насчитываться сотни тысяч), а также по соображениями безопасности пользователей (система DNS не контролируется оператором связи, и он не может предотвратить утечку из нее данных). В крупных корпоративных сетях отображение с помощью ENUM используется, так как оба ее недостатка здесь не проявляются — масштаб не такой и безопасность можно обеспечить своими силами.

Вместо обращения к системе ENUM операторы применяют традиционную практику связи первых цифр телефонного номера с определенным оператором связи, например, если номер начинается с цифр +7 903, то всем операторам мобильной связи известно, что этот номер принадлежит компании Билайн. Соответствие старших цифр номеров телефонов операторам связи и, следовательно, их именам доменов в системе DNS хранится в домашних базах HSS операторов и доступно серверам регистрации системы IMS.

Вторая задача всегда решается с помощью обращения к DNS. Поиск сервера нужного типа в определенном домене включает три последовательных запроса к DNS. Рассмотрим эту процедуру на примере поиска запрашивающего типа в домене operator2.net. В исходном состоянии известны только тип сервера — запрашивающий — и имя домена.

Первым посылается запрос типа NAPTR (Name Authority Pointer) с аргументом operator2.net. По этому запросу система DNS возвращает список сервисов, которые поддерживаются серверами домена operator2.net, например sip, h.323, email. Запись NAPTR в свое время создал администратор домена operator2.net. Но одного такого запроса недостаточно, теперь нужно найти имя сервера, который в домене operator2.net отвечает за сервис sip. Поэтому сервер регистрации делает новый запрос к DNS другого типа — SRV (Service), с аргументами sip и operator2.net, и этот запрос возвращает имя i-cscf2.operator2.net. И наконец, стандартный запрос A (IPv4) или AAAA (IPv6) с этим именем возвращает IP-адрес запрашивающего сервера i-cscf2.operator2.net домена operator2.net. Обратите внимание, что в запросе SRV не указывался тип сервера, и интерпретация ответа SRV как имя запрашивающего сервера полностью полагается на тот факт, что оператор при формировании записи SRV поместил туда имя запрашивающего сервера. Формат записи типа SRV не дает возможности определить тип сервера, а только то, что сервер участвует в предоставлении определенного сервиса, например сервиса sip. Обычно операторы помещают в записи SRV для сервиса sip имя запрашивающего сервера, так как сервер этого типа нужен для нахождения сервера регистрации определенного домена.

Установление соединения

На примере двух доменов, изображенных на рис. 24.13, рассмотрим, каким образом системы IMS этих доменов устанавливают соединение между двумя телефонами, когда пользователь +733333333@operator1.net инициирует соединение с пользователем +711111111@operator2.net, набирая на своем телефоне номер +711111111.

Будем считать, что оба телефона прошли регистрацию в своих домашних доменах, то есть пользователь +711111111@operator2.net с помощью своего прокси-сервера P-CSCF2 зарегистрирован в одном из серверов регистрации домена operator2.net, а пользователь +733333333@operator1.net с помощью своего прокси-сервера P-CSCF1 зарегистрирован в единственном сервере регистрации домена operator2.net (оператор этого домена решил обойтись одним сервером регистрации).

После того как пользователь набрал номер +711111111, его телефон посылает запрос INVITE +711111111 через *свой* прокси-сервер *своему домашнему* серверу регистрации s-cscf.operator1.net. Сервер регистрации дополняет телефонный номер именем домена вызываемого абонента, найдя это имя по отображению первых нескольких цифр телефонного номера на доменное имя в своей домашней базе HSS или воспользовавшись услугой ENUM DNS, в результате чего запрос теперь стал выглядеть как INVITE 711111111@operator2.net.

Сервер регистрации s-cscf.operator1.net теперь знает полное имя вызываемого пользователя, но оно ничего не говорит о его текущей точке присоединения, то есть какой прокси-сервер его в данный момент обслуживает. Он может находиться в любом домене, и об этом знает только сервер регистрации домашнего домена operator2.net вызываемого пользователя. Так как в домене operator2.net может быть несколько серверов регистрации, то нужно сначала найти запрашивающий сервер этого домена, который должен знать, какой из серверов регистрации домена отслеживает перемещения пользователя +711111111@operator2.net.

Поиск запрашивающего сервера происходит с помощью системы DNS и включает запросы NAPTR/SRV/A, описанные выше. Результатом запросов является IP-адрес запрашивающего сервера i-cscf2.operator2.net домена operator2.net, и по нему сервер регистрации s-cscf.operator1.net обращается к запрашивающему серверу i-cscf2.operator2.net.

Запрашивающий сервер i-cscf2.operator2.net знает, какой сервер регистрации закреплен за пользователем +711111111@operator2.net, поэтому он возвращает имя этого сервера, например s-cscf21.operator2.net. Получив ответ, сервер регистрации пользователя 733333333+@operator1.net переправляет запрос INVITE +711111111@operator2.net серверу регистрации пользователя +711111111@operator2.net, — а тот знает текущее положение в сети своего пользователя, то есть его прокси-сервер p-cscf2@operator2.net.

В результате домашний сервер регистрации пользователя +711111111@operator2.net пересылает этому прокси-серверу приглашение INVITE +711111111@operator2.net, после чего прокси-сервер пересылает его телефону +711111111. Дальнейшее развитие событий зависит от вызываемого пользователя. Если он, услышав звонок, решит на него ответить, то его согласие в виде ответа OK пойдет назад по цепочке, но уже без обращения к системе DNS, так как последовательность посещенных узлов при передаче сообщения INVITE была записана в заголовок этого сообщения, так что обратный путь для сообщения OK известен.

Из ответа OK телефон пользователя +733333333@operator1.net узнает IP-адрес, который ему нужно использовать в прямой передаче голосовых данных в сообщениях протокола RTP телефону 711111111@operator2.net. В системе SIP этим адресом является IP-адрес вызываемого телефона, но операторы сетей LTE обычно избегают прямого общения телефонов пользователей между собой. Они соединяют телефоны пользователей через промежуточный узел системы IMS, который называется медиашлюзом, и это его IP-адрес возвращает сообщение OK. Этот промежуточный медиашлюз является единственным узлом, принадлежащим слою пользовательских данных системы IMS. Все остальные ее узлы работают только с протоколом сигнализации SIP, то есть принадлежат слою управления. Необходимость прохождения пользовательского голосового трафика через медиа-шлюз вызвана несколькими причинами: требованиями учета проходящего трафика для его оплаты, требованиями безопасности, а также управлением сессиями — переадресацией звонков, постановкой звонка на ожидание и т. п.

IP-мобильность и SIP-мобильность

В приведенном описании двух основных операций в системе IMS — регистрации и установления сессии — мы не затрагивали вопросы управления IP-мобильностью. Это не упущение, а отражение того факта, что управление IP-мобильностью в сети LTE на основе эстафетной передачи почти полностью экранирует систему IMS от проблем, связанных с перемещением устройства пользователя между сотами: если IP-адрес при этих перемещениях остается неизменным, то сервер регистрации считает, что устройство не сменило домен регистрации. Единственной ситуацией, когда система IMS должна реагировать на перемещение устройства пользователя между сотами является смена шлюза PGW и соответственно смена IP-адреса. В этом случае устройство должно пройти новую регистрацию в системе IMS.

SIP-мобильность происходит на другом уровне — уровне доменов IMS. Система IMS не требует новой регистрации, если устройство не перешло под контроль другого прокси-сервера, что случается при смене доменов.

Новая регистрация происходит и в других ситуациях — при роуминге, при выключении-включении устройства пользователя, а также она может быть вызвана истечением срока регистрации.

Мобильные сотовые сети 5G

Новый взгляд на роль мобильных сетей

Концепция сетей LTE предполагала эволюционное развитие технологий и услуг, без резких революционных изменений. В течение довольно длительного времени так и происходило, что позволяло называть новые релизы стандартов LTE стандартами 4-го поколения. Вносились усовершенствования в радиосеть доступа за счет различных новшеств (увеличение числа антенн для параллельного обмена данными MIMO, агрегирование нескольких частотных диапазонов в общий с шириной полосы в 100 МГц на оператора и др.), повышающих скорость приема и передачи данных. Но все эти изменения были недостаточными, чтобы назвать усовершенствованные сети сетями следующего поколения. Они по-прежнему назывались сетями 4-го поколения, с различными приставками: LTEA (Advanced LTE), LTEB и LTEC.

Новшество, послужившее основанием для использования термина «сети 5-го поколения» (5G), состояло в виртуализации сети, в превращении ее в программируемую сеть, использующую все последние достижения в области облачных центров данных, виртуализации сетевых функций NFV и программно-определяемых сетей SDN.

Причиной, приведшей к такой трансформации мобильной сети, стало стремление к революционному расширению набора услуг, оказываемых мобильной сетью. При этом имеются в виду услуги сетей всех типов: телефонных и компьютерных, стационарных и мобильных, публичных и корпоративных. Эти услуги должны охватывать не только традиционных пользователей мобильных сетей (людей с телефонами и планшетами), но и неодушевленных пользователей: автоматизированные промышленные устройства и установки, транспортные средства и всевозможные объекты интернета вещей, оснащенные датчиками и исполнительными механизмами. Беспроводной доступ к таким услугам для новых типов

«пользователей» подходит очень хорошо, не ограничивая свободу передвижения устройств проводами. Для достижения этой цели архитектура мобильной сети неминуемо должна измениться, став платформой, поддерживающей весь жизненный цикл существования услуги: разработку услуги с помощью программных средств в коллективной среде (подобной GitHub), отладку и тестирование услуги на реальном оборудовании мобильной сети, запуск услуги в эксплуатацию и поддержание во время эксплуатации, прекращение предоставления услуги с безопасным удалением всех ее элементов из сети. Предполагается, что в описанных выше процессах разработки и предоставления услуг сетей 5-го поколения принимают участие различные заинтересованные стороны:

- конечные пользователи разных типов: индивидуальные, корпоративные, промышленные объекты и т. п.;
- провайдеры конечных услуг различных типов: провайдеры телефонных услуг, скоростного доступа к интернету, услуг интернет-телевидения, услуг управления промышленными объектами и т. п.;
- разработчики конечных услуг: программисты, интеграторы;
- провайдеры услуг инфраструктуры: центров данных, виртуальных сетей;
- оператор сети — организация, отвечающая в целом за функционирования сети как платформы для создания и предоставления услуг.

Альтернативой программируемости сети может быть только создание в ней независимых физических сетей, каждая из которых служит для оказания услуги определенного вида. По этому пути шли разработчики сетей предыдущих поколений, и яркий пример такого подхода — существование отдельных сетей для предоставления услуг телефонии и доступа к интернету в сетях 2G и 3G. Но если такой подход успешно работал для случая двух разнородных услуг, то он вряд ли является перспективным, если планируется оказание десятков услуг разного типа, при том что точно не известно, какие новые услуги понадобится предоставлять в ближайшем будущем. Таким образом, необходимы принципиально новые решения.

Первый стандарт, в котором концепции нового типа мобильных сетей было решено называть сетями 5-го поколения, появился в июне 2018 года. С тех пор технология сетей 5G непрерывно развивалась.

Области применения сетей 5G

В первом стандарте сетей 5G были определены услуги в трех основных областях (помимо услуги мобильного телефонного соединения):

- **Улучшенный мобильный скоростной доступ к интернету** (Enhanced Mobile Broadband, eMBB). Эта услуга отличается от аналогичной услуги 4G/LTE более высокими скоростями передачи данных, более высокой мобильностью пользователей, более высокой плотностью соединений в зависимости от сценария ее предоставления: город, офис, сельская местность, борт самолета и т. д. Примерами средних скоростей (так называемая «скорость на основании опыта пользователя») такого доступа являются: 100 Мбит/с при приеме и 50 Мбит/с при передаче для города, сельской местности (пешеходы) и в быстро движущихся поездах (до 500 км/ч) и автомобилях (до 250 км/ч). Для плотно населенных городских территорий (аэропорт, стадион, торговый центр) должна обес-

печиваться скорость при приеме до 300–500 Мбит/с. В офисах ожидается скорость приема до 1 Гбит/с. Для пассажиров самолета будет обеспечиваться скорость 1,2 Гбит/с на самолет (разделяемая между всеми пассажирами). Приведенные здесь требования относятся к практически ожидаемым скоростям, а теоретические максимумы сети 5G равны 20 Гбит/с для скорости приема и 10 Гбит/с для скорости передачи.

- **Критические коммуникации** (Critical Communications, **CC**) и **сверхнадежная связь с низкой задержкой** (Ultra Reliable and Low Latency Communications, **URLLC**). Ряд областей применения требуют от мобильной сети передачи данных с очень низкой задержкой и очень высокой степенью надежности. Это управление промышленными объектами в реальном времени, управление беспилотными автомобилями и аналогичными транспортными средствами, управление роботами, оказание срочной медицинской помощи. Для этих целей требования к скорости передачи не такие высокие, как у услуг доступа в интернет, но зато очень высоки требования к величине задержки данных: не более 1–5 мс для менее требовательных приложений (оказание срочной медицинской помощи) и не более 0,5 мс для более требовательных (управление промышленным процессом или беспилотным автомобилем). Во многих случаях потребуется очень высокая надежность соединения (до 99,9999 %), так как прерывание может привести к катастрофическим последствиям. Перенос серверов в пограничный домен сети 5G в соответствии с тенденцией пограничных вычислений может существенно сократить задержки.
- **Массивный интернет вещей** (Massive Internet of Things, **MIoT**). Интернет вещей населен огромным количеством объектов различной природы, требующих обмена данными для выполнения своих функций. В конце 2025 года количество пользователей-«вещей» превосходило количество «живых» пользователей интернета почти в 4 раза (20 миллиардов против 5,6 миллиарда). Специфические требования этой области применения сетей 5G: низкое энергопотребление сетевых адаптеров этих устройств, которое должно позволить им работать без подзаряда до 10 лет; широкая территория покрытия. Требования к скорости обмена данными для этой области в основном умеренные, в районе 10 Мбит/с, при этом для некоторых сценариев требования к задержке могут быть достаточно жесткими, от 1 до 5 мс.

В последующих выпусках стандартов появились новые области применения сетей 5G. Наиболее значимые из них:

- **Расширенная реальность** (eXtended Reality, XR). Эта область включает виртуальную реальность, то есть полную иллюзию реальности, не имеющей связи с реальным миром, и дополненную реальность, то есть наложение цифровой информации на изображение реального мира в режиме реального времени, как правило, с использованием смартфонов или умных очков. Эта область применения ужесточает требования к задержке радиосети (1–4 мс) и джиттеру (1–2 мс).
- **Устройства с ограниченной функциональностью** (Reduced Capabilities, **RedCap**). Такие устройства занимают промежуточное положение между смартфонами (полная функциональность) и устройствами массового интернета вещей (очень низкое энергопотребление, низкие скорости передачи данных). Примерами таких устройств являются нательные камеры, промышленные датчики, медицинские приборы. Сеть 5G выделяет таким устройствам более узкую полосу частот (то есть меньше ресурсных блоков), а это сокращает энергопотребление устройства, и более низкие скорости передачи

данных — максимум 150–200 Мбит/с нисходящего потока и максимум 50–100 Мбит/с восходящего.

- **Автомобили и дроны**. Эта услуга имеет формальное название V2X (Vehicle to Everything) — «транспортное средство — все, что угодно». Услуга является разновидностью услуги «Критические коммуникации» и рассчитана на предоставление связи беспилотным транспортным средствам — автомобилям и дронам, которым требуются еще большая надежность связи и меньшие задержки.

Виртуализация сети 5G

Виртуализация сети — главное отличие сетей 5G от мобильных сетей предыдущих поколений, которое должно помочь решить проблему совмещения в одной сети различных услуг, предъявляющих различные и часто противоречивые требования к ее характеристикам. Под виртуализацией понимается создание в сети различных логических сетей над общей разделяемой физической инфраструктурой. Известно несколько примеров технологий, основанных на виртуализации физических ресурсов. Так, технология виртуальных локальных сетей (VLAN) позволяет разделить общую физическую сеть, построенную на коммутаторах, на отдельные, непосредственно не взаимодействующие логические сети, которые затем можно объединить маршрутизаторами и создать требуемую топологию сети, которую, в свою очередь, можно изменять логически, не прибегая к реконфигурации физических связей в сети. Другим широко распространенным примером виртуализации является техника виртуальных машин (VM), за счет которой физические ресурсы одного компьютера разделяются между несколькими виртуальными компьютерами, каждый из которых работает под управлением отдельной ОС и потребляет отведенную ему долю физических ресурсов — процессорных циклов или ядер, памяти, диска и производительности сетевых интерфейсов. Еще одним примером виртуализации является технология виртуализации сетевых функций NFV.

В сетях 5G виртуальной единицей является слайс сети.

Слайс (slice — срез, ломтик) сети 5G — набор элементов сети, специализирующихся на предоставлении определенного сервиса или типа сервисов и образующих виртуальную сеть 5G.

Так, в сети может существовать один слайс, предоставляющий услуги интернета вещей, другой слайс — для поддержки «классических» услуг телефонии, третий — для услуг транспорта и т. д. Разные слайсы отвечают разным требованиям, предъявляемым к производительности и параметрам QoS, или же отличаются ориентацией на определенные типы пользователей: например, пользователи публичных служб безопасности (скорая помощь или пожарная бригада), «пользователи» интернета вещей. Слайс «нарезает» физическую сеть «из конца в конец», охватывая все транспортные домены мобильной сети — радиосеть, транзитная сеть, магистральная сеть, а также центры данных, в которых расположены серверы сети, на которых работают приложения, выполняющие функции слоя управления. Различные слайсы могут использоваться одновременно и взаимодействовать друг с другом. В зависимости от особенностей своего бизнеса оператор решает, как много слайсов ему надо внедрить в своей сети и какие функции нужно разделять между слайсами.

Каждый слайс характеризуется в терминах QoS (задержка, джиттер), а также скоростью передачи данных, надежностью соединения и максимально допустимой скоростью пере-

мещения мобильных устройств. Набор характеристик слайса может быть предопределен в стандарте 3GPP или же определен самим оператором. В первом стандарте сетей 5G 2019 года имеется три типа предопределенных слайсов:

- Слайс 1-го типа предназначен для реализации услуг улучшенного широкополосного доступа в интернет (eMBB).
- Слайс 2-го типа — для реализации услуги сверхнадежных коммуникаций с низкой задержкой (URLLC).
- Слайс 3-го типа — услуга массового интернета вещей (MIoT).

В последующих версиях стандартов были добавлены еще несколько предопределенных слайсов, например для услуг беспилотных транспортных средств V2X.

На основе предопределенных слайсов оператор связи может определять свои собственные слайсы — например, может быть определен слайс для управления беспилотными автомобилями, который будет подслайсом слайса 2-го типа. Над уровнем слайсов располагается уровень приложений, оказывающих специфические услуги конечным пользователям. В терминах семиуровневой модели OSI слайс можно рассматривать как реализацию нижних уровней этой модели, включая транспортный. Последний является очень гибким, предоставляя не два типа транспортного сервиса (UDP и TCP), а гораздо больше и с гораздо более точно определенными характеристиками.

В сети 5G имеется функция выбора слайса NSSF (Network Slice Selection Function), с помощью которой приложение, работающее на устройстве пользователя, сможет выбрать необходимый слайс. Устройство пользователя может обращаться к услугам сразу нескольких слайсов. У устройства могут быть заданы правила отображения некоторого приложения на слайс определенного типа. Например, приложение «Видео по требованию» будет в соответствии с этими правилами выбирать слайс 1-го типа, а приложение слежения за температурой в доме — 3-го типа.

Для организации таких сложных систем, как слайсы, и гибкой среды для разработки и предоставления широкого спектра услуг в сети 5G потребовалось применение новых подходов и технологий. Поиск этих подходов и технологий происходил на основе опыта, приобретенного провайдерами облачных сервисов, который показал, что наиболее эффективные решения основаны на повышении программируемости сети. К наиболее перспективным технологиям, обеспечивающим программируемость сети и тем самым наделяющим мобильные сети возможностями быстрой адаптации к требованиям новых типов пользователей и новых типов услуг, были отнесены технологии SDN и NFV (см. главу 16).

Архитектура сети 5G

Ввиду того что в сети 5G широко используются технологии виртуализации, ее архитектуру достаточно сложно представить на одной структурной схеме, так как между ее виртуальными элементами могут существовать разнообразные связи, создающие сложную паутину линий.

Сеть 5G, как и ее предшественники, состоит из нескольких транспортных сетей (рис. 24.14):

- сетей доступа нескольких типов;
- магистральной сети 5GC (5G Core);

- транзитной сети, соединяющей сеть радиодоступа с магистральной сетью (Transport Network).

Транзитная сеть аналогична по функциям сети агрегирования трафика крупных глобальных сетей (см. главу 19), которая мультиплексирует трафик от многочисленных сетей доступа в высокоскоростные каналы магистральной сети.

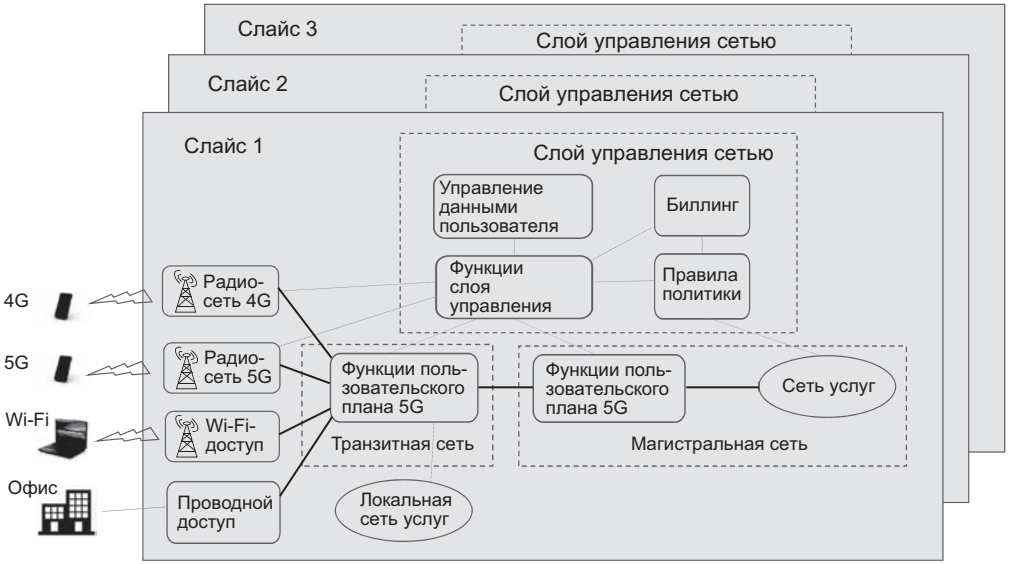


Рис. 24.14. Обобщенная архитектура сети 5G

Функциональные элементы сети 5G делятся на две категории. К первой относятся функции пользовательского плана, то есть занимающиеся продвижением IP-пакетов в транзитной и магистральной сетях (маршрутизаторы, коммутаторы, файерволы). Вторую категорию образуют функции плана управления, в которые, наряду с протоколами плана управления стека TCP/IP, входят специфические функции управления мобильной сетью (управление мобильностью, учетными данными пользователя, биллинг и др.). Два новых элемента — сеть услуг и локальная сеть услуг — являются сетями центров данных провайдеров услуг сети 5G. В этих сетях находятся серверы приложений, которые, собственно, и реализуют специфические услуги этой сети. Локальная сеть услуг приближена к пользователям для уменьшения времени реакции сети до 0,5–5 мс, что необходимо для критических коммуникаций и некоторых услуг интернета вещей.

Однако на этом схожесть архитектур сетей 4G и 5G заканчивается. Архитектуру сети 5G нельзя описать в стиле архитектуры сети 4G с жесткими связями между функциональными узлами сети. Например, шлюзы SWG и PWG всегда связаны с узлом управления мобильностью MME.

Вместо описания жестких интерфейсов между функциями архитектура 5G описывает программный интерфейс доступа к каждой функции, оставляя вопрос связей между функциями открытым. Предполагается, что функция A может воспользоваться услугами функции B, запросив эту услугу с помощью соответствующего программного интерфейса

(API). Программный интерфейс базируется на протоколе HTTP (REST API). Такой способ представления архитектуры в гораздо большей степени соответствует виртуальной природе сети 5G, в которой связи между блоками могут гибко изменяться в зависимости от назначения слайса.

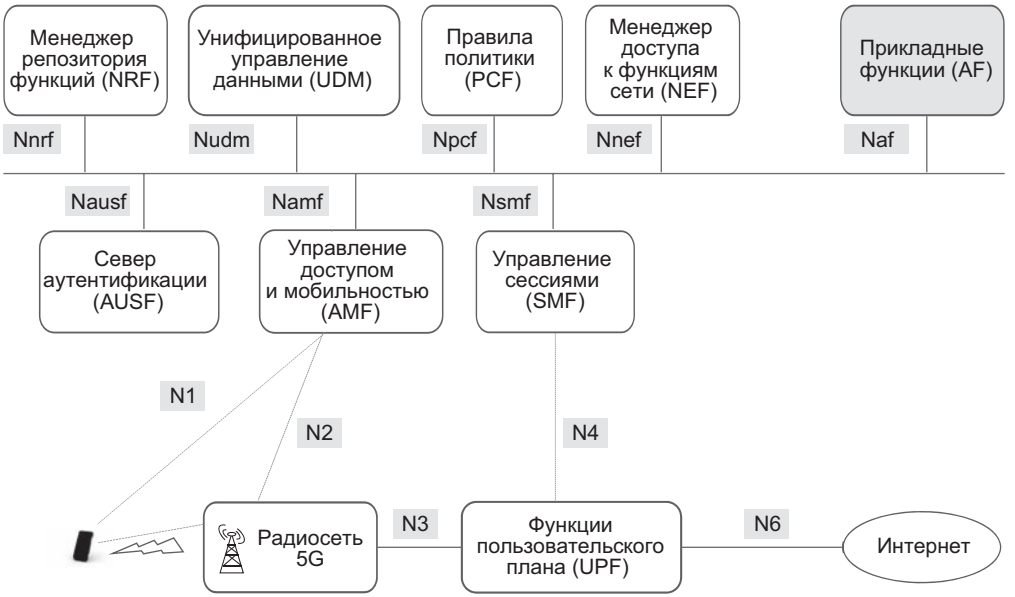


Рис. 24.15. Представление архитектуры 5G, основанное на услугах

На рис. 24.15 можно увидеть, что функции слоя управления (функция сервера аутентификации — Authentication Server Function, AUSF, функция управления доступом и мобильностью, функция управления сессиями — Session Management Function, SMF, и ряд других) не связаны друг с другом жесткими связями. Вместо этого показана их связь с некоторой общей шиной обмена сообщениями, с помощью которой они могут пользоваться услугами друг друга. Возле символа каждой функции показан ее программный интерфейс. Например, Nausf — программный интерфейс функции сервера аутентификации AUSF. Если функции управления доступом и мобильностью AMF нужно выполнить аутентификацию пользователя, то она может вызвать функцию сервера аутентификации AUSF, направив ей запрос в соответствии с правилами интерфейса Nausf через общую шину обмена сообщениями.

В наборе функций слоя управления имеются и служебные функции, поддерживающие взаимодействие между остальными функциями. К таким функциям относится *функция менеджера репозитория функций*: этот репозиторий представляет собой базу данных описаний всех функций системы, что помогает переносу некоторой услуги из сети одного оператора в сеть другого оператора, так как услуга может проверить, имеются ли в наличии нужные функции для ее реализации.

Особое место в этой архитектуре занимают ***прикладные функции*** (Application Functions, AF), реализующие логику некоторой пользовательской услуги. Прикладная функция также может вызывать другие функции для выполнения некоторой работы, но при этом вызывать их непосредственно она может только тогда, когда оператор сети «одобрил» прикладную функцию, то есть признал ее безопасной. Прикладная функция, разработанная третьей стороной и не одобренная оператором, может вызвать сетевую функцию только через посредника — ***менеджера доступа к функциям сети***.

Основанное на услугах (взаимных услугах сетевых функций) представление архитектуры сети 5G позволяет гибко строить цепи функций в стиле NFV, не ограничиваясь жесткой структурой связей между ними.

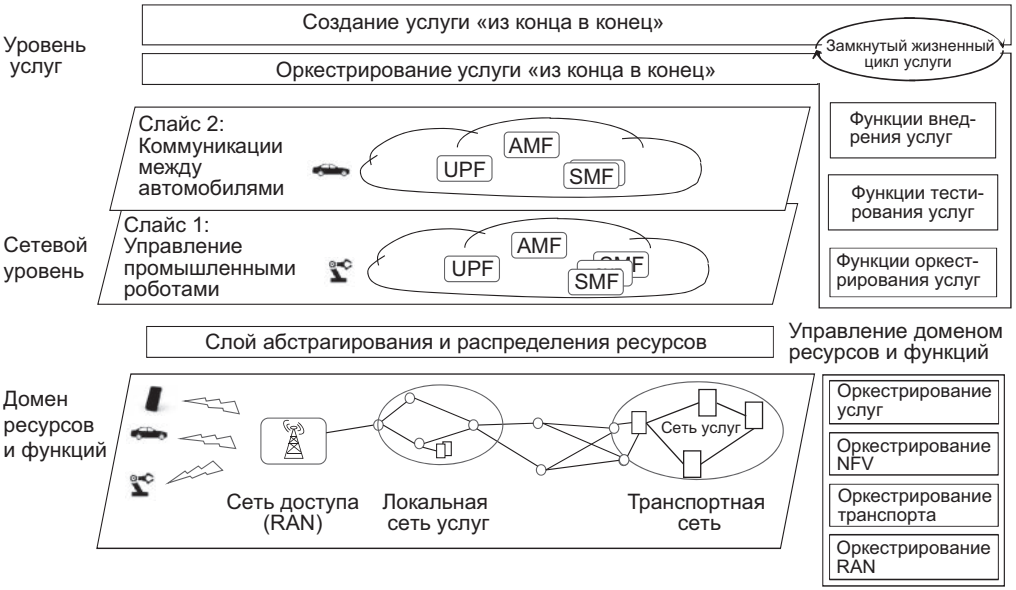


Рис. 24.16. Взаимоотношения физических ресурсов, сетевых слайсов и услуг в сети 5G

На рис. 24.16 показан пример сети 5G, иллюстрирующий взаимоотношения между физическими ресурсами сети, сетевыми слайсами и услугами «из конца в конец» для конечных пользователей сети, предложенный рабочей группой по разработке архитектуры сети 5G ассоциации 5GPPP (5G Public Private Partnership). На диаграмме можно увидеть, что физические ресурсы сети отображаются в два сетевых слайса, отвечающих требованиям областей управления автомобилями и промышленными роботами. На основе этих слайсов создаются конкретные услуги для названных типов пользователей. В правой части диаграммы показаны оркестраторы различного типа, позволяющие быстро создавать виртуальные элементы сети: транспортные сети пользовательского слоя, слайсы и услуги для конечных пользователей.

(S) *Сравнение процедур создания туннелей в сетях LTE и 5G*

Новое радио

У сети радиодоступа 5G появились задачи и цели, которых не было у сети радиодоступа LTE. В то время как у сети радиодоступа LTE главной задачей было обеспечение качественной передачи трафика одного типа — трафика услуги широкополосного доступа к интернет (eMBB в терминах 5G), то сеть радиодоступа 5G должна передавать одновременно трафик разных слайсов — eMBB, URLCC, MIoT, V2X с существенно разными требованиями к скорости передачи данных и задержкам в обслуживании.

Для достижения такого баланса в обслуживании трафика различных классов QoS в радиосети 5G применены новые технологии и приемы. Некоторые из них уже введены в поздних версиях стандартов сетей LTE (LTE Advanced), но не нашли широкого применения. В сетях 5G они развиты и усовершенствованы. В то же время в радиосети доступа 5G имеются и совсем новые технологии, что, вероятно, и дало повод называть эту сеть доступа «Новое радио» (New Radio, NR).

Нужно подчеркнуть, что сеть радиодоступа 5G NR может использоваться и сама по себе, то есть без остальной части сети 5G (магистрали 5GC и транзитной сети 5G). Например, сеть 5G NR может использоваться как сеть радиодоступа сети 4G/LTE. Другим примером применения сети 5G NR может служить ее работа в сети дронов (см. главу 27).

Рассмотрим сначала те свойства сети 5G, которые являются улучшенными свойствами сети LTE/4G.

Унаследованные свойства 5G NR:

- Тот же стек протоколов радиосреды PCPD-RRC-RLC-MAC-PhY с тем же распределением функций между уровнями.
- Распределение пропускной способности радиосреды ресурсными блоками с 12 частотными подканалами протоколом MAC на стороне базовой станции.
- Мультиплексирование OFDMA — но в сетях 5G оно применяется не только в нисходящем направлении, но и в восходящем, то есть оно теперь работает и в смартфоне при отправке данных базовой станции. Недостаток этого метода мультиплексирования — высокое отношение пиковой мощности передатчика к средней — уже не рассматривается производителями смартфонов как существенный. Прогресс в области технологии микросхем радиопередатчиков позволил его преодолеть.
- Применение систем антенн MIMO, но в сетях 4G на практике обычно ограничивались схемами MIMO для устройств UE 1 × 2 и 2 × 2, а в сетях 5G телефоны должны поддерживать схему 8 × 8 для приема и 4 × 4 для передачи. Кроме того, в базовых станциях сети 5G широко применяются массивные многопользовательские системы MU-MIMO со схемами 64 × 64 и 128 × 128. Техника формирования луча стала в сетях 5G намного совершеннее.

Рассмотрим новые свойства радиосети доступа 5G NR.

Новый диапазон частот миллиметровых волн mmWave (24–52 ГГц)

Это дает сети 5G NR большие возможности в повышении скорости передачи данных (вспомните эмпирическое правило, говорящее, что ширина полосы сигнала не должна пре-

вышать 5 % от значения несущей частоты). В подобном диапазоне мобильные телефоны еще не работали — он используется в основном для спутниковой и радиорелейной связи, в радарах и сканерах. Основным достоинством этого диапазона, называемого также диапазоном FR2, является его бóльшая частотная емкость по сравнению с гораздо более «населенными» диапазонами частот ниже 6 ГГц (диапазон FR1). В разных странах она разная, но, например, только в диапазоне со средней частотой 27 ГГц, по оценкам специалистов, для работы сетей 5G можно использовать частотные полосы с суммарной емкостью около 3,5 ГГц.

Нужно отметить, что термин «технология mmWave» используется в сегодняшней литературе по телекоммуникационным сетям в двух значениях. Первое значение относится к технологии 5G NR, работающей в диапазоне FR2, который является поддиапазоном всего диапазона миллиметровых волн. Второе значение относится к любой радиотехнологии, работающей в миллиметровом диапазоне волн, от 30 ГГц до 300 Ггц, например к радиорелейным линиям «точка-точка», к спутниковым системам «спутник — наземная станция».

Гибкое планирование распределения ресурсных блоков

В сети 5G планировщик MAC-уровня поддерживает три очереди для трафика трех разных услуг (eMBB, URLCC и MioT).

Для очереди трафика доступа к интернету eMBB он производит планирование не для каждого тайм-слота — это было бы расточительно для трафика этого типа с требованиями к задержке в 10–15 мс. Вместо этого он выполняет планирование один раз в 2–4–8 тайм-слотов и старается заполнить имеющиеся ресурсные блоки как можно полнее, назначая несколько последовательных во времени тайм-слотов некоторому устройству пользователя в одном цикле планирования.

Для очереди сверхнадежного трафика, чувствительного к задержкам URLCC, он выполняет планирование для каждого тайм-слота. При этом он может использовать укороченные во времени ресурсные блоки в 2–7 символов, чтобы быстрее их отправить. При приходе срочных данных URLCC запланированная заранее передача нескольких ресурсных блоков трафика eMBB в последовательных тайм-слотах может прерываться, и данные URLCC будут помещены в ресурсный блок, ранее предназначенный данным eMBB. То есть планировщик MAC выполняет обслуживание очередей с абсолютными приоритетами (см. главу 5).

Применение малых сот

Применение миллиметровых волн позволяет существенно повысить скорость передачи данных, но оно имеет и свои недостатки. В главе 22 упомянуты проблемы распространения волн этого диапазона (при наличии в воздухе водяных капель или даже тумана). Кроме того, чем выше частота, тем хуже проникает сигнал через препятствия, а значит, для приема такого сигнала в помещении понадобится внешняя антенна.

Чем выше частота, тем быстрее убывает и энергия сигнала с расстоянием от источника. При частоте сигнала 26 ГГц и мощности передатчика максимальный радиус покрытия составляет 377 м (для сравнения: радиус покрытия сигнала частотой 3,5 ГГц, которая также

используется «Новым радио», равен 2 км). Небольшой радиус покрытия требует применения малых сот размером от нескольких десятков до сотен метров, а также большого количества антенн в базовых станциях этих сот, компенсирующих плохое прохождение сигнала через препятствия. В сетях 5G могут применяться соты различных размеров: метро (сотни метров, >250 пользователей), микро (несколько сотен метров, 100–250 пользователей), пико (десятки метров, 15–100 пользователей) и фемто (десятки метров в жилых домах и офисных зданиях, 4–30 пользователей).

Для сот малых размеров проблема интерференции сигналов от различных базовых станций становится очень острой (базовые станции находятся на близких расстояниях друг от друга), и бороться с ней с помощью сложных методов настройки антенн MIMO становится малоэффективно. Поэтому в сотах малого размера применяется метод повторного использования частот.

Эстафетная передачи при двойном подключении

В сетях 5G эстафетная передача при переходе устройства пользователя из соты одной базовой станции в соту другой базовой станции происходит без потери связности с сетью, если эти станции соединены прямым интерфейсом и базовые станции и устройство пользователя поддерживают режим двойного подключения (Dual Connectivity, DC).

Режим DC состоит в том, что устройство пользователя при присоединении к сети присоединяется к двум сотам и распараллеливает свой поток пакетов на два между базовыми станциями этих сот. Сначала устройство пользователя присоединяется к соте с лучшим сигналом в ходе стандартной процедуры. Базовая станция этой соты становится главной станцией MN (от Master Node). Если устройство пользователя информирует станцию MN о том, что оно получает сильные сигналы от другой базовой станции, то станция MN решает, что ее нужно сделать вторичной станцией SN (от Secondary Node). Главная станция MN по прямому интерфейсу сообщает об этом решении вторичной станции SN и передает ей всю информацию об устройстве пользователя, необходимую для работы устройства в соте станции SN: контекст устройства в радиосреде, контекст безопасности, параметры QoS. Станция MN также уведомляет устройство пользователя о необходимости подключения ко вторичной станции SN, сообщая ему необходимые для этого параметры соты. Устройство пользователя входит в синхронизм с базовой станцией SN и присоединяется к ее соте (упрощенная процедура, без аутентификации), а также уведомляет MN о своей готовности работать с двумя сотами, после чего все три участника схемы DC начинают передавать трафик параллельно по двум связям.

На рис. 24.17 показаны пути параллельных потоков в восходящем направлении: поток 1 передается устройством пользователя главной станции MN, а поток 2 передается сначала вторичной станции SN, которая затем передает его главной станции MN. Главная станция MN объединяет пакеты потоков 1 и 2 в общий поток и отправляет его в магистральную сеть 5GC через шлюз сети 5G.

При передаче трафика устройства пользователя в обратном, нисходящем направлении расщепление пакетов на два параллельных пути выполняется главной базовой станцией MN, а сбор в единый поток происходит на конечном пункте маршрута — устройстве пользователя. В обоих случаях вторичная базовая станция SN играет вспомогательную роль промежуточного релейного узла.

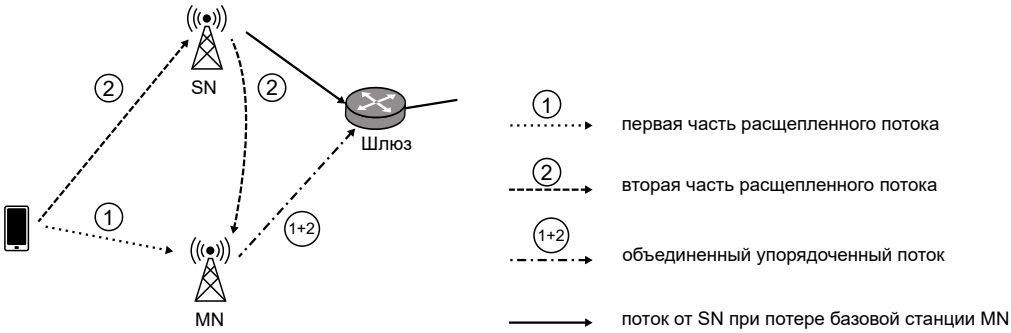


Рис. 24.17. Передача данных при двойном подключении

Естественно, возникает проблема: IP-пакеты могут приходить на узел сбора потоков — станцию MN или устройство пользователя — не в том порядке, в каком их отправлял источник потока. Если их не упорядочить, то протоколы верхних уровней, такие как, например, TCP, будут рассматривать неупорядоченность пакетов как признак проблем в работе сети и пытаться исправить ситуацию путем замедления трафика или повторной пересылки пакетов. Большинство решений, использующих распараллеливание трафика между несколькими линиями связи, расщепляют поток пакетов или кадров таким образом, что поток одной и той же сессии протокола верхнего уровня всегда направляется по одной и той же линии связи (расщепление на основе адреса назначения). В этом случае последовательность пакетов потока не нарушается.

В сетях 5G техника DC работает по другому принципу — поток расщепляется на уровне пакетов, невзирая на адрес назначения. Поэтому на точку сбора потока пакеты могут приходить в неверной последовательности, но их переупорядочиванием там занимается протокол PDCP. Как вы помните, этот протокол работает под протоколом IP в стеке радиосреды и нумерует передаваемые им пакеты IP. Упорядочивание пришедших по разным путям IP пакетов — это его новая обязанность, которую он выполняет тогда, когда активизируется механизм DC на главной базовой станции MN и устройстве UE. И если протокол PDCP справляется с этой функцией, то протоколы уровня IP и выше просто не замечают того факта, что на каком-то участке сети IP-пакеты передавались в неверном порядке.

Теперь посмотрим, как механизм DC помогает сети 5G выполнять эстафетную передачу. Возможны несколько сценариев такой передачи.

Сценарий 1. Главная базовая станция MN не меняется, меняется только вторичная станция SN. В этом случае станция MN замечает, что сигнал от старой вторичной станции стал ослабевать, в то же время сигнал от некоторой базовой станции стал заметно усиливаться. Главная базовая станция MN принимает решение заменить старую вторичную базовую станцию на новую. Главная станция руководит процессом перехода устройства пользователя от старой базовой станции к новой, посылая всю нужную информацию для этого устройству пользователя и новой базовой станции. Во время потери соединения со старой устройство UE сохраняет связность с главной базовой станцией, то есть цель двойного подключения достигнута — эстафетная передача проходит по схеме «make-before-break» («создание перед разрывом») и устройство пользователя никогда во время передач не теряет связи с сетью.

Сценарий 2А. Главная базовая станция MN меняется, при этом старая вторичная базовая станция остается в области покрытия устройства пользователя во время перехода к новой главной базовой станции. В этом случае на время разрыва связи с главными базовыми станциями старая вторичная базовая станция используется для передачи всего трафика между устройством пользователя и сетью, используя свою связь со шлюзом. После активизации связи устройства пользователя с новой главной базовой станцией последняя может получить замену вторичной базовой станции, найдя новую станцию с более сильным сигналом.

Эстафетная передача проходит по схеме «make-before-break».

Сценарий 2Б. Главная базовая станция MN меняется, при этом связь со старой вторичной базовой станцией также теряется. Это обычный вариант эстафетной передачи по схеме «break-before-make» с потерей связи с сетью на небольшой период времени.

Двойное подключение DC приводит к «бесшовной» эстафетной передаче не во всех случаях, но при передачах между географически близкими сотами она часто работает и помогает сохранять связность сети и устройства пользователя, что особенно важно для критических приложений класса URLCC.

Развитие сетей 5G

Новые версии стандартов развивают функциональные возможности сетей 5G в различных направлениях.

- *Повышение скорости передачи данных*. Продолжающийся рост трафика интернета (см. рис. 1.6) требует от компьютерных сетей постоянного наращивания их скорости и емкости. Одной из возможностей для повышения скорости передачи данных в радиосети является расширение диапазона используемых частот, и разработчики стандартов сетей 5G пошли по этому пути — они добавили к диапазону FR2 новый поддиапазон FR22 52–71 ГГц. Этот поддиапазон включает нелицензируемый в глобальном масштабе диапазон частот с центром 60 ГГц, что упрощает развертывание сетей.
- *Поддержка «простых» устройств*, таких как камеры наблюдения, разнообразные датчики и т. п. Это свойство, называемое NRLight, обеспечивается за счет выделения таким устройствам радиоканалов с более узкой полосой частот, упрощенной конструкцией антенн и более простой схемой кодирования данных.
- *Поддержка неземных сетей* (NonTerrestrial Networks, NTN). Такие сети используют спутники и высоко летящие дроны в качестве базовых станций, создавая гибридную «наземно-воздушную сеть». Особенности этой технологии описаны далее в этой главе в разделе «Спутниковые сети».
- *Возможность построения Ad Hoc-сетей* на основе технологии 5G NR Sidelink. Особенности этой версии 5G кратко описаны в главе 27 «Сети дронов».

Наряду с усовершенствованием технологии 5G ведущие телекоммуникационные исследовательские центры начали работы над созданием технологии следующего поколения — технологии 6G. Она обещает быть еще более быстрой и объединить реальный и виртуальный мир за счет поддержки виртуального присутствия пользователей сети в любой точке мира с помощью беспроводных соединений и элементов искусственного интеллекта.

Спутниковые сети

Типы спутниковых сетей

Идея использовать искусственный спутник Земли для создания линий связи родилась задолго до запуска в 1957 году первого такого спутника Советским Союзом. Писатель-фантаст Артур Кларк продолжил дело Жюля Верна и Герберта Уэллса, которым удалось описать множество технических изобретений еще до их появления. Кларк в 1945 году описал геостационарный спутник, который висит над одной точкой экватора и обеспечивает связью большую часть территории Земли.

Сегодня спутники используются для оказания самых разнообразных услуг своим пользователям на земле, в небе и в космосе. Имея хороший обзор и область покрытия, спутники используются для сбора разнообразных данных — о состоянии атмосферы Земли и океанов, картографических данных, данных о состоянии районов бедствия, шпионской информации и многих других типов данных. Спутники незаменимы для навигации — морской, авиационной и наземной, предоставляя своим клиентам точные координаты в пространстве, например, с помощью наиболее распространенной системы этого типа GPS. Ретрансляция телевизионных сигналов и радиостанций также является одной из наиболее востребованных услуг спутников.

Из всего спектра услуг нас, естественно, интересуют телекоммуникационные услуги спутниковых сетей, они же услуги связи. Первый спутник, запущенный Советским Союзом в годы холодной войны, обладал очень ограниченными телекоммуникационными возможностями — он только передавал радиосигнал «бип-бип», извещая мир о своем присутствии в космосе. Однако успех России в космосе подхлестнул усилия Америки, и в 1962 году она запустила первый телекоммуникационный спутник Telstar-1, который поддерживал 600 телефонных каналов.

Со времени запуска первого телекоммуникационного спутника прошло уже более 60 лет, и функции спутника как телекоммуникационного узла, естественно, усложнились. Сегодня спутник может играть роль телефонного коммутатора и коммутатора/маршрутизатора компьютерной сети, предоставляя услуги доступа к интернету и голосовой и видеосвязи. Для этого аппаратура спутников взаимодействует не только с наземными станциями, но и между собой, образуя прямые космические беспроводные линии связи, на основе которых строятся **спутниковые сети**. Принципиально техника передачи радиосигналов в космосе и на Земле не отличается, однако у спутниковых сетей есть и очевидная специфика — часть узлов такой сети постоянно с высокой скоростью перемещается в космическом пространстве на большом расстоянии от остальных наземных узлов.

Коммуникационные технологии спутниковых систем часто являются закрытыми, фирменными. Например, так обстоит дело со спутниковыми системами Iridium, Inmarsat, Starlink. В то же время появляются международные стандарты, определяющие те или иные аспекты функционирования спутниковых коммуникаций. Например, спецификация 5G NTN (5G Non Terrestrial Network, внеземная сеть 5G) описывает расширение технологии мобильных сотовых сетей для спутников и дронов.

Спутники и спутниковые системы в качестве телекоммуникационных сетей могут быть классифицированы по различным характеристикам, при этом наиболее важными являются:

- частотный диапазон;
- высота орбиты;
- используемые беспроводные технологии связи;
- топология связей.

В зависимости от сочетания этих, а также некоторых других характеристик, спутниковая система может предоставлять тот или иной набор услуг.

Частотные диапазоны

Для спутниковой связи союз ITU выделил несколько частотных диапазонов: L (1–2 ГГц), S (2–4 ГГц), C (4–8 ГГц), Ku (12–18 ГГц), Ka (26,5–40 ГГц) и V (50–80 ГГц).

Первые три диапазона относятся к низкочастотным, а три последних — к высокочастотным диапазонам.

Низкочастотные диапазоны обладают такими преимуществами, как высокая степень проникновения через препятствия (стены, деревья и т. п.), а также меньшая зависимость затухания сигнала от расстояния, которое, как вы помните, пропорционально квадрату частоты. Но эти диапазоны обладают существенным недостатком — из-за их сравнительной узкой полосы и населенности другими типами беспроводных сетей и устройств (например, мобильными сотовыми сетями, радарами и бытовыми приборами) скорость передачи данных в них низкая: до 1 Мбит/с в L-диапазоне и до 100 Мбит/с в C-диапазоне.

У высокочастотных диапазонов основным преимуществом является гораздо более широкая полоса частот, выделяемая спутниковой системе (так как сами диапазоны гораздо шире и они гораздо менее населены). Поэтому скорости передачи данных здесь существенно выше — до 300 Мбит/с в диапазоне Ku, до 1 Гбит/с в диапазоне Ka и до 10 Гбит/с в диапазоне V. Но для достижения таких скоростей спутниковым системам приходится применять специальные технические приемы, так как у высокочастотных сигналов затухание выше примерно на 20 дБ. Для преодоления этого недостатка в высокочастотных диапазонах применяются антенны MIMO с узконаправленным управляемым лучом, позволяющим сконцентрировать энергию луча в точке приема. К недостаткам высокочастотных диапазонов относится также их зависимость от погодных условий.

Для связей «спутник — спутник» кроме радиосвязи используется также лазерная связь, которая требует прямой видимости между антеннами передатчика и приемника — условие, которое для случая связи между спутниками часто выполняется. Лазерная связь может обеспечить очень высокие скорости обмена данными, до 10–100 Гбит/с.

Орбиты

Спутники отличаются высотой орбиты над Землей. Существует три группы орбит (рис. 24.18):

- геостационарная орбита (Geostationary Orbit, GEO) — 35 863 км;
- средневысотная орбита (Medium Earth Orbit, MEO) — 5000–15 000 км;
- маловысотная орбита (Low Earth Orbit, LEO) — 100–1000 км.

Спутники всех трех групп орбит используют передачу данных во всех частотных диапазонах, в зависимости от услуг, которые они предоставляют.

Геостационарный спутник GEO «висит» на большой высоте 35 863 км над определенной точкой экватора, в точности следуя скорости вращения Земли. При таком положении спутника его зона покрытия составляет около трети земной поверхности, поэтому трех таких спутников достаточно для предоставления сервисов по всему земному шару. GEO-спутник обычно имеет большое количество антенн и радиопередатчиков для обслуживания многочисленных клиентов, попадающих в его зону покрытия.

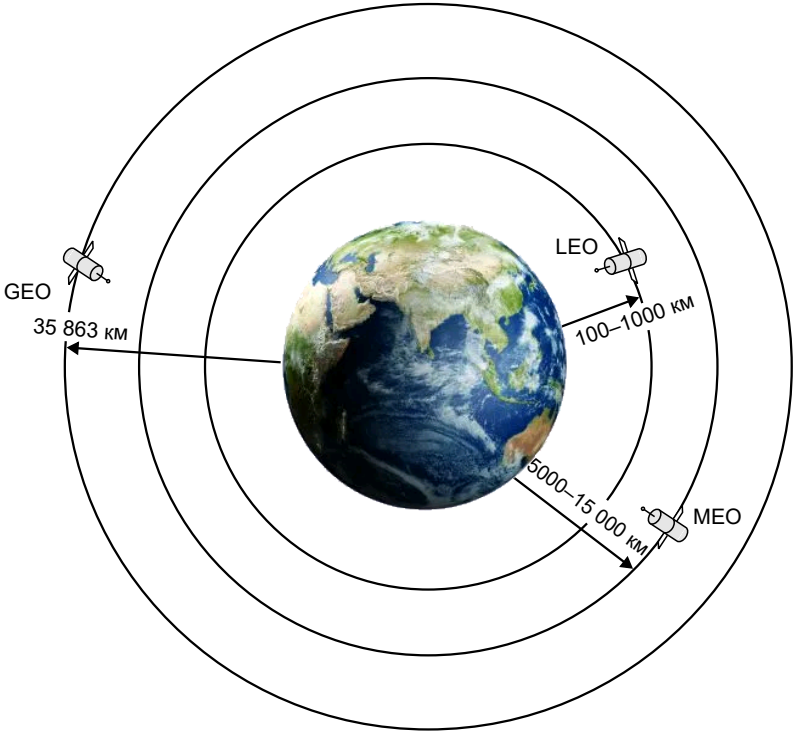


Рис. 24.18. Типы орбит спутников

Наиболее очевидным недостатком геостационарного спутника является большая задержка сигнала, обусловленная большим удалением спутника от поверхности Земли. Большое расстояние GEO-спутника от поверхности Земли приводит к большим задержкам распространения сигнала: односторонняя задержка (OWD) от 230 до 280 мс, а двусторонняя (RTT) — около 500 мс. При использовании спутника для передачи разговора или телевизионного диалога возникают неудобные паузы, мешающие нормальному общению, но широковещание телевизионных неинтерактивных программ очень хорошо сочетается с характеристиками GEO-спутника. Эти же паузы препятствуют нормальной работе протокола TCP, который был разработан в расчете на то, что задержки в передаче пакетов и квитанций ACK не превышают 100 мс. Поэтому провайдеры IP-сервисов на основе GEO-спутников предпринимают специальные меры для своих клиентов:

- Наземные станции используются как прокси-узлы, которые буферизуют пакеты клиентов и отвечают от имени конечного узла квитанциями ACK. Это сокращает задержки в ответах до приемлемых 50 мс.

- Используется специальная версия протокола TCP — SCPS-TP (Space Communications Protocol Standards — Transport Protocol) с настраиваемыми тайм-аутами и выборочной политикой повторных передач пакетов.
- На канальном уровне работают процедуры восстановления кадров в стиле HARQ мобильных сотовых сетей LTE/5G (см. раздел «Стек протоколов сети радиодоступа» выше).

В частотном диапазоне Ka (~30 ГГц) затухание для спутника GEO составляет около 200 dB, что требует применения передатчика высокой мощности и антенны с управлением лучом.

Примером системы GEO-спутников является Inmarsat. Это британская компания спутниковой связи, предоставляющая глобальные услуги мобильной связи, включая телефонную связь и передачу данных пользователям по всему миру. Она эксплуатирует систему геостационарных спутников и предлагает решения для обеспечения связи в различных секторах, включая морской, авиационный и государственный.

Среднеорбитальные спутники MEO обеспечивают диаметр покрытия на Земле от 10 000 до 15 000 км и одностороннюю задержку распространения сигнала 50–100 мс.

Спутники MEO занимают промежуточное положение между спутниками GEO и LEO не только по положению орбиты, но и по характеристикам. При достаточно большой области покрытия (обычно требуется 10–30 спутников для создания региональной или глобальной области покрытия) такие спутники вносят приемлемые задержки, а затухание сигнала на частоте Ka составляет на 10 дБ (то есть в 10 раз) меньше, чем у спутников GEO.

Наиболее известной услугой, предоставляемой спутниками этого класса, является глобальная система навигации (Global Positioning System, GPS). GPS — это система определения текущих координат пользователя на поверхности Земли или в околоземном пространстве. GPS состоит из 24 спутников — это то минимальное число спутников, которое необходимо для стопроцентного покрытия территории Земли. Спутники GPS летают на орбите высотой около 20 000 км. Помимо спутников в систему GPS входит сеть наземных станций слежения за ними и неограниченное количество пользовательских приемников-вычислителей.

По радиосигналам спутников GPS-приемники пользователей устойчиво и точно определяют координаты; для этого на поверхности Земли приемнику необходимо принять сигналы как минимум от трех спутников. Погрешности не превышают десятков метров. Этого вполне достаточно для решения задач навигации подвижных объектов (самолеты, корабли, космические аппараты, автомобили и т. д.).

Однако, несмотря на хороший баланс характеристик, среднеорбитальные спутниковые системы MEO проигрывают низкоорбитальным системам LEO борьбу на рынке телекоммуникационных услуг — во многом из-за сравнительно большой задержки сигнала.

Низкоорбитальные спутники LEO. Из-за того что спутники этого типа расположены ближе к Земле их передатчики могут иметь меньшую мощность, они могут быть оснащены антеннами меньших размеров, а их сигналы распространяются с меньшей задержкой (около 20–25 мс). Кроме того, спутники LEO имеют меньшие габариты и их легче запускать.

Основной недостаток — малая площадь покрытия, диаметр которой составляет всего около 2000–8000 км. Период оборота такого спутника вокруг Земли составляет 1,5–2 часа, а время видимости спутника наземной станцией — всего 20 минут. Это значит, что постоянная связь с помощью низкоорбитальных спутников может быть обеспечена, только когда на

орбите находится достаточно большое их количество, обычно *несколько тысяч* — и такое массовое производство значительно их удешевляет.

В том случае, когда спутник LEO выполняет роль базовой станции сотовой коммуникационной сети, такая сеть *всегда будет мобильной*, даже когда пользовательское оборудование сети на Земле неподвижно и располагается в фиксированных точках поверхности. Это происходит из-за того, что сота, создаваемая отдельным спутником, — зона покрытия отдельного спутника — движется по поверхности Земли, причем с большой скоростью. Так что даже фиксированное устройство пользователя постоянно перемещается между сотами, и сеть постоянно должна выполнять процедуру эстафетной передачи.

Еще одним недостатком низкоорбитальных спутников является значительные отклонения частоты сигналов из-за **эффекта Доплера** (см. раздел «Распространение электромагнитных волн» в главе 23). Низкоорбитальные спутники имеют большую радиальную составляющую скорости (проекцию вектора скорости спутника на прямую, соединяющую спутник с наблюдателем) относительно неподвижной точки на поверхности Земли. Например, спутник, находящийся на орбите высотой 550 км, имеет максимальную радиальную скорость 7,4 км/с, которая достигается в нижних точках наблюдения спутника, то есть при появлении спутника из-за горизонта и при опускании его за линию горизонта, а в зените эта скорость равна нулю. Так как сдвиг частоты сигнала из-за эффекта Доплера зависит от частоты сигнала, то на низких частотах, например 1,9 ГГц, максимальный сдвиг частоты для такого спутника будет равен ± 48 кГц при скорости изменения сдвига (от зенита до горизонта) ± 50 кГц/с, на высоких частотах этот сдвиг существенно больше: в диапазоне Ku (12 ГГц) он равен примерно 300 кГц со скоростью изменения 330 кГц/с, а в диапазоне Ka (20 ГГц) — 500 кГц со скоростью изменения частоты 550 кГц/с. Очевидно, что приемопередатчики низкоорбитальных спутников должны иметь специальные механизмы отслеживания положения спутника над горизонтом относительно определенного пользователя и компенсации динамического сдвига частоты. Кроме того, из-за эффекта Доплера ширина частотных каналов в диапазонах Ku и Ka должна быть увеличена по сравнению с каналами наземных беспроводных систем, обычно до 10–50 МГц на пользователя с шириной подканалов (если применяется мультиплексирование OFDM) 0,5–2 МГц.

Если основным назначением геостационарных спутников является ретрансляция радио и телепередач и дальняя связь, то низкоорбитальные спутники рассматриваются как важное средство высокоскоростного доступа к интернету и поддержания мобильной связи. Ниже мы рассмотрим две системы низкоорбитальных спутников, Iridium и Starlink.

Iridium

Первой коммерческой системой низкоорбитальных спутников, покрывшей 100 % поверхности Земли, была система Iridium — группа из 66 спутников, запущенная в 1997 году. Примерно половина первоначальных спутников Iridium были выведены на орбиту российской ракетой «Протон».

Проект Iridium имел весьма амбициозную цель — создать всемирную спутниковую сеть, предоставляющую услуги мобильной телефонной связи в любой точке земного шара. В конце 80-х, когда этот проект стартовал, еще не существовало такой плотной сети сот мобильной телефонии, как сегодня, так что коммерческий успех казался обеспеченным.

Спутники Iridium первого поколения покрывали всю поверхность земного шара, вращаясь по 6 орбитам, проходящим через полюсы земли. На каждой орбите высотой 780 км находилось по 11 спутников, передатчики которых работали на частоте 1,6 ГГц с полосой пропускания 10 МГц. Спутники связаны между собой каналами высокочастотной связи в диапазоне Ka, этот же диапазон использовался для связи спутников с наземными базовыми станциями.

Телефонный трафик передавался на основе техники коммутации каналов. За счет многократного использования частот система Iridium поддерживала 253 440 телефонных каналов, организуя системы скользящих по поверхности Земли сот. Для пользователей системы Iridium основным видом услуги являлась телефонная связь и передача данных со скоростью 2,4 Кбит/с.

К сожалению, коммерческие успехи Iridium оказались очень скромными, и через два года своего существования компания обанкротилась. Расчет на мобильных телефонных абонентов оказался неверным — к моменту начала работы наземная сеть сотовой связи уже покрывала большую часть территории развитых стран. А услуги по передаче данных со скоростью 2,4 Кбит/с не соответствовали потребностям пользователей конца XX века.

Система Iridium пережила банкротство в самом начальном периоде своей деятельности, выжила и сегодня предоставляет своим пользователям (около трех миллионов подписчиков) услуги трех типов: телефонную связь для удаленных районов, высокоскоростной доступ к интернету в удаленных районах (до 700 Кбит/с) и низкоскоростной доступ к интернету вещей (до 128 Кбит/с) во всемирном масштабе.

Система Iridium по-прежнему состоит из 66 спутников на тех же орбитах, но это уже четвертое поколение спутников, и их программное и аппаратное обеспечение существенно изменилось. Коммуникационная сеть перешла на стандартные протоколы: стек TCP/IP на сетевом и транспортном уровнях, а на физическом уровне применяется специальный протокол, NB-IoT, разработанный для связи мобильных сетей LTE с устройствами интернета вещей, нетребовательных к скорости передачи данных и задержкам кадров (около 1 секунды), но потребляющих очень мало энергии.

Система Iridium нашла свою нишу, обслуживая устройства интернета вещей и пользователей в удаленных районах.

Starlink

Система низкоорбитальных спутников Starlink на сегодня является наиболее крупной спутниковой сетью (и по количеству активных спутников, и по количеству подписчиков), предоставляющей услуги высокоскоростного доступа к интернету. На конец 2025 года в системе Starlink насчитывалось свыше 8700 активных спутников и около 7 миллионов подписчиков. Диаметр области покрытия отдельного спутника составляет около 200–1000 км, что приводит к необходимости выполнения эстафетной передачи между сотами каждые 15 секунд. Скорость доступа находится в диапазоне от 100 до 300 Мбит/с, а задержка — в границах 20–40 мс. Областью покрытия системы спутников Starlink является почти вся поверхность земли (за исключением приполярных областей), однако из-за ограничений, накладываемых регуляторами, услуги предоставляются не во всех странах, в которых они технически возможны (в России не предоставляются). Спутники Starlink передают данные клиентов на базовые наземные станции (около 150 во всем мире), которые связаны проводными линиями с магистральными сетями телекоммуникационных провайдеров.

Как видно из характеристик Starlink, цель ее создания кардинально отличалась от цели первого поколения системы Iridium (создание всемирной мобильной телефонной сети), что, конечно, отразилось на основных идеях и принципах ее устройства. Перед создателями системы Starlink ставилась задача построения всемирной системы предоставления *высокоскоростных услуг доступа к интернету*. При этом система Starlink должна была быть способной конкурировать с наземными провайдерами таких услуг.

Начальная идея создания такой всемирной летающей паутины была обнародована в 2015 году, когда глава компании SpaceX Илон Маск объявил об инициативе по созданию системы низкоорбитальных спутников, обеспечивающих бесплатный высокоскоростной доступ в интернет по всему миру. План состоял в запуске 700 недорогих микроспутников весом 110 кг на орбиту высотой 1200 км. Для обеспечения высокой скорости передачи данных планировалось использовать высокочастотные диапазоны Ka и Ku. Этот план начал воплощаться в жизнь, и в 2021 году первые 1584 спутника стали обслуживать клиентов, летая на орбите высотой 550 км. Эти спутники были распределены по 24 орбитальным плоскостям, по 66 спутников в каждой плоскости с наклоном плоскости в 53°, так что они действительно окутывали Землю плотной паутиной (рис. 24.19).

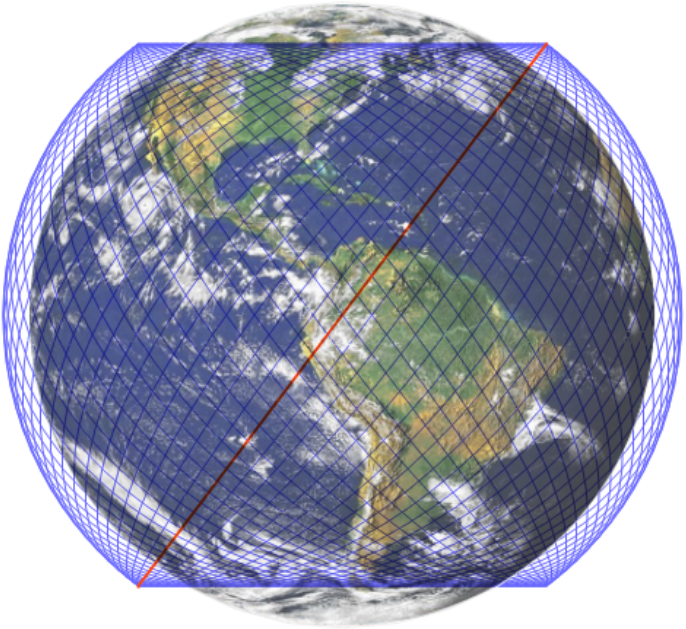


Рис. 24.19. Орбиты спутников Starlink

Источник: https://commons.wikimedia.org/wiki/File:Starlink_SpaceX_1584_satellites_72_Planes_22each.png Автор: Lamid58

За прошедшие годы количество спутников увеличилось, а их орбиты стали разнообразнее: по высоте — от 350 до 550 км, по углу наклона — 53°, 70°, 97,6°, по количеству спутников в каждой плоскости — от 20 до 72. Такая высокая плотность спутников (расстояние между плоскостями 250–500 км, а расстояние между спутниками в одной плоскости — 500–

1500 км) обеспечивает видимость нескольких десятков спутников одновременно почти в каждой точке земного шара, а значит, и высокую надежность связи за счет избыточности спутников.

Спутники Starlink, несмотря на свою относительно короткую историю, уже насчитывают четыре поколения, отличающиеся весом и, естественно, оборудованием и функциональными возможностями. Спутник первого поколения весил около 260 кг, а спутник последнего четвертого поколения — 2000 кг.

Независимо от поколения спутник Starlink оснащен:

- *ионно-криогенными двигателями*, которые после отделения спутника от ракеты-носителя выводят его на заданную орбиту, а также осуществляют навигационные маневры для избежания столкновения с препятствиями;
- *солнечными батареями*, имеющими размер до 50 м^2 (в спутниках последнего поколения), они обеспечивают электроэнергией передатчик мощностью 10–30 Ватт;
- *антенной* для связи с наземными терминалами пользователей и базовой станцией; антенна спутника выполнена в виде плоской фазированной решетки площадью до 25 м^2 с электронным управлением формирования луча.

Система спутников Starlink обслуживает пользователей с фиксированным местом расположения. Мобильность в том виде, в котором она существует в мобильных сотовых сетях, не поддерживается. Пользователь снабжается *терминалом*, состоящим из плоской фазированной решетки меньшего, чем на спутнике, размера (59 × 51 см), работающей в режиме 4 × 4 MIMO. Мобильность пользователей в принципе представляет собой сложную проблему для низкоорбитальных спутниковых систем, так как они обязаны поддерживать мобильность летающих с высокой скоростью базовых станций сети с выполнением эстафетной передачи между их сотами каждые 10–15 минут. Поэтому дополнительная мобильность пользователей требует дополнительных процедур в стеке протоколов спутниковой сети и, возможно, дополнительного оборудования у пользователя. Пока услуги Starlink разрешают пользователю перемещаться только между несколькими фиксированными точками присутствия, а для пользователей, которые хотят получать услуги в процессе движения, необходимо приобретать специальное терминальное оборудование.

Приемопередатчики спутников Starlink отслеживают положение спутника относительно фиксированного пользователя и динамически корректируют частоту передаваемого сигнала для компенсации значительного эффекта Доплера в диапазонах частот Ka и Ku, так что терминальное оборудование пользователя почти не испытывает смещения частоты. Аналогично, приемник спутника производит коррекцию сдвига частоты принимаемого от пользователя сигнала.

Спутник Starlink первого поколения был в основном отражателем высокочастотного сигнала от терминального оборудования пользователя — он его принимал и переправлял одной из наземных базовых станций. Оборудование для выполнения такой функции было несложным, что и позволяло применять небольшие и легкие спутники. Из-за отсутствия межспутниковых связей первое поколение спутников Starlink не могло обслуживать пользователей на море и в труднодоступных местностях — если нет наземной станции, то нет и возможности передавать трафик в проводную магистральную сеть. Но уже со второго поколения спутники Starlink стали выполнять функции базовых станций беспроводной сети и обмениваться информацией между собой, используя сначала радиосвязи, а потом — лазерные связи, обеспечивающие высокие скорости обмена, от 10 до 100 Гбит/с.

Таким образом, последнее поколение спутников Starlink представляет собой «летающую» телекоммуникационную сеть, в которой узлы сети являются комбинацией маршрутизатора и базовой станции мобильной сотовой сети.

Starlink является закрытой коммерческой системой, как и большинство существующих спутниковых систем. Поэтому многие детали о внутренней организации системы спутников Starlink как телекоммуникационной сети остаются неизвестными. Известно, что в сети Starlink работает стек протоколов TCP/IP. Это видно по IP-адресам пользовательских терминалов (публичным IPv6- и частным IPv4-адресам для индивидуальных пользователей и публичным IPv4-адресам для корпоративных пользователей). Опыт пользователей показывает, что протоколы TCP и UDP нормально работают через сеть Starlink, без изменения настроек тайм-аутов TCP или приложений. Однако детали протокола маршрутизации между спутниками и наземными станциями неизвестны.

Прямая связь с телефоном

В 2022 году появилась новая технология **5G NTN** (Non Terrestrial Networks) — 5G для внеземных сетей. В этих сетях в качестве базовых станций используются спутники GEO/MEO/LEO, а также так называемые высотные платформы **HAPS** (High Altitude Pseudo Satellite), которые могут представлять собой медленно летящие дроны на высоте 18–22 км, то есть выше самолетов, но ниже LEO-спутников. Применение дронов в качестве базовых станций, летающих намного ближе к пользователям, чем спутники, позволяет использовать в качестве устройств пользователя обычные 5G-смартфоны, устраняя необходимость в специальных спутниковых телефонах и терминалах. Услуга сети 5G NTN получила название **«Прямая связь с телефоном»** (Direct-to-phone).

По сравнению со стандартной версией 5G были сделаны следующие изменения:

- Расстояния между подканалами OFDMA увеличены для борьбы с эффектом Доплера. Наряду со стандартным подканалом 15 кГц допускается использовать подканалы 60, 120, 240, 480 и 960 кГц. Каналы шириной 60 кГц и выше предназначены для диапазона частот FR2 (mmWave). Увеличение ширины частотного подканала пропорционально уменьшает длительность тайм-слота OFDMA, длительность тайм-слота подканала 960 кГц равна 0,0156 мс — в 64 раза меньше стандартного для наземных сетей 5G тайм-слота 1 мс, так что частота принимаемого терминалом сигнала смещается из-за эффекта Доплера за время тайм-слота в 64 раза меньше, что упрощает работу корректирующих схем. Это изменение ориентировано на работу 5G NTN в низкоорбитальных спутниках.
- В процедуре HARQ, обеспечивающей надежную доставку транспортных блоков на MAC-уровне, увеличены временные интервалы для устойчивой работы на геостационарных спутниках.
- Процедура эстафетной передачи изменена, она теперь стала *предсказуемой*. Это означает, что в отличие от наземной сотовой сети, в которой причиной смены соты было непредсказуемое перемещение пользователя, в спутниковой сети смену соты можно предвидеть, так как она меняется в соответствии с устойчивой траекторией полета спутников. Поэтому сеть заранее готовится к смене соты для определенного пользователя, зная траекторию спутника, его точное положение от системы GPS, предсказуемое ослабление сигнала от текущей соты (спутника) и усиление сигнала от соседней соты

(спутника). Эта же информация используется для управления направленным лучом антенны MU-MIMO.

Существуют следующие режимы работы технологии 5G NTN:

Прозрачный: спутники или дроны просто принимают сигналы от устройств пользователя и передают их без обработки и изменения наземному шлюзу или наземной базовой станции. Этот режим предназначен для применения в ситуациях, когда в пределах области покрытия устройств пользователя нет базовых станций — в труднодоступных или безлюдных местностях, при ликвидации катастроф, когда нужно обеспечить мобильную связь.

Регенеративный, когда спутник или дрон выполняют функции базовой станции. Этот режим существует в двух вариантах.

- В первом из них связи между летающими базовыми станциями отсутствуют. Этот вариант соответствует наземной сети 5G с отсутствующими прямыми интерфейсами между базовыми станциями. Взаимодействие между летающими базовыми станциями происходит только через наземные шлюзы и узлы магистральной сети.
- Во втором варианте связи между летающими базовыми станциями существуют, причем допускается создание ячеистой сети летающих базовых станций с передачей данных через промежуточные узлы, маршруты для которых выбираются с помощью протокола маршрутизации. Второй вариант более надежен и допускает работу сети даже в том случае, когда наземные шлюзы находятся за пределами зон покрытия спутников/дронов. Протокол маршрутизации для этого варианта сети 5G NTN пока не определен.

ГЛАВА 25 Технологии Wi-Fi и Bluetooth

История создания и основные характеристики Wi-Fi

Сети и оборудование стандарта **IEEE 802.11**, также известные под названием **Wi-Fi** — по имени консорциума Wi-Fi[1] Alliance, который занимается вопросами совместимости и сертификации оборудования стандартов IEEE 802.11, — занимают лидирующие позиции в мире беспроводных локальных сетей. Целью разработчиков технологии Wi-Fi было создание беспроводного аналога сетей Ethernet, и история показала, что этой цели им достичь удалось. Сегодня трудно найти квартиру, дом, офис, аэропорт, где бы не работала сеть Wi-Fi, давая компьютерам, ноутбукам и мобильным телефонам возможность доступа к интернету. По данным Wi-Fi Alliance, в 2024 году в мире использовалось около 21 миллиарда устройств Wi-Fi. Растущие размеры интернета вещей (IoT) должны были добавить к этому количеству еще 188 миллионов в 2025 году.

Первый стандарт сети Wi-Fi был создан в 1997 году рабочей группой IEEE 802.11. В нем были определены основные принципы организации сети, которые сохранились и во всех последующих версиях этого стандарта:

- Технология включает два уровня — физический и канальный, который в свою очередь делится на подуровень доступа к среде MAC и подуровень логического управления каналом LLC. То есть структура стека протоколов Wi-Fi соответствует структуре стека протоколов других технологий канального уровня, разработанных комитетом IEEE 802.
- Узлы адресуются с помощью 48-битных MAC-адресов, которые, как и MAC-адреса узлов Ethernet, являются глобально уникальными, при этом их уникальность обеспечивается производителями оборудования Wi-Fi.
- Узлы сети Wi-Fi работают в нелицензируемых диапазонах частот ISM (2,4 ГГц, 5 ГГц и 6 ГГц), так что для установки такой сети в доме или офисе получать разрешение от регулятора не нужно.
- Сеть строится на основе базовой структуры — так называемой сети BSS (Basic Service Set), которая представляет собой соту с точкой доступа (базовой станцией) с областью покрытия до несколько сот квадратных метров, что достаточно для многих применений. На основе сетей BSS могут создаваться более сложные, многосегментные сети или сети с ячеистой топологией.

Сети Wi-Fi создавались для пользователей, которые перемещаются в основном пределах соты базовой станции. Поэтому процедура эстафетной передачи, применяемая в мобиль-

---

[1] Wi-Fi — термин был введен в целях маркетинга по аналогии с популярным термином Hi-Fi, обозначающим высокую точность воспроизведения звука аппаратурой.

ных сотовых сетях, не была предусмотрена в первых стандартах Wi-Fi. Со временем ситуация стала меняться, и в больших сетях Wi-Fi (корпоративных, сетях аэропортов и т. п.) возникла потребность в использовании многих сот и поддержании перехода пользователя из одной соты в другую. Поэтому процедуры перехода из соты в соту были стандартизованы и постепенно стали приближаться по скорости и качеству к процедурам эстафетной передачи в мобильных сотовых сетях.

Большой вклад в усовершенствование технологии Wi-Fi наряду с рабочей группой вносит Wi-Fi Alliance. Этот консорциум производителей оборудования Wi-Fi разрабатывает как сами стандарты (которые потом рассматриваются и ратифицируются рабочей группой IEEE 802.11), так и программы сертификации изделий на соответствие стандартам, что очень важно для совместимости.

Сети Wi-Fi работают на основе методов мультиплексирования с расширением частоты, таких как FHSS, DSSS (более ранние версии стандарта), а также OFDM и OFDMA (более поздние версии), что сближает их с сетями радиодоступа мобильных сотовых сетей 4G/5G.

В сетях Wi-Fi применяются различные методы модуляции радиосигналов, такие как BPSK, QPSK, 16-QAM, 64-QAM, 256-QAM и 1024-QAM. Гибкое управление частотными подканалами (при использовании OFDM и OFDMA) и выбор типа модуляции позволяет адаптировать скорость передачи данных к состоянию радиосреды и расстоянию между взаимодействующими узлами сети.

Стандарты сетей Wi-Fi постоянно развивались и совершенствовались на протяжении почти 30-летней истории их существования. Скорости передачи данных в ранних версиях стандарта 802.11 были небольшими — от 1 Мбит/с до 11 Мбит/с, но в более поздних версиях они возросли до сотен Мбит/с, а в последней версии Wi-Fi 7 максимальная теоретическая скорость равна 46 Гбит/с. Следующая версия Wi-Fi 8 ожидается в 2028 году.

Сети Wi-Fi могут строиться в рамках различных архитектур. Они различаются наличием инфраструктуры (сеть с точкой доступа) или ее отсутствием (одноранговые сети AdHoc), количеством точек доступа в инфраструктурных вариантах, а также рядом других характеристик. Одной из самых распространенных на практике архитектур сетей Wi-Fi является так называемая базовая архитектура, с которой мы и начнем изучение сетей этого типа.

Архитектура сети BSS

Элементы базовой сети BSS

Архитектура **базовой сети Wi-Fi**, называемой в стандарте **BSS** (Basic Service Set), предусматривает наличие у сети инфраструктуры: сеть состоит из одного центрального инфраструктурного узла — **точки доступа** (Access Point, **AP**) и некоторого количества конечных узлов, называемых **станциями** (Station, **STA**).

Область покрытия точки доступа, называемая **сотой**, является также областью покрытия сети BSS. Предполагается, что все конечные узлы сети могут обмениваться кадрами с точкой доступа AP, но не обязательно непосредственно между собой, так как две станции могут находиться по разные стороны от точки доступа и «не доставать» друг до друга (об эффекте скрытой станции читайте в главе 22).

На рис. 25.1 показан вариант сети BSS в котором несколько ноутбуков и мобильных телефонов представляют собой станции, обслуживаемые точкой доступа AP. Все узлы сети попадают в область покрытия точки доступа AP (область покрытия показана овалом), области покрытия конечных узлов не показаны, но предполагается, что все они охватывают точку доступа AP, так что все узлы сети имеют двусторонние физические и логические связи с точкой доступа AP. То есть сеть BSS представляет собой *одношаговую* (single hop) сеть.

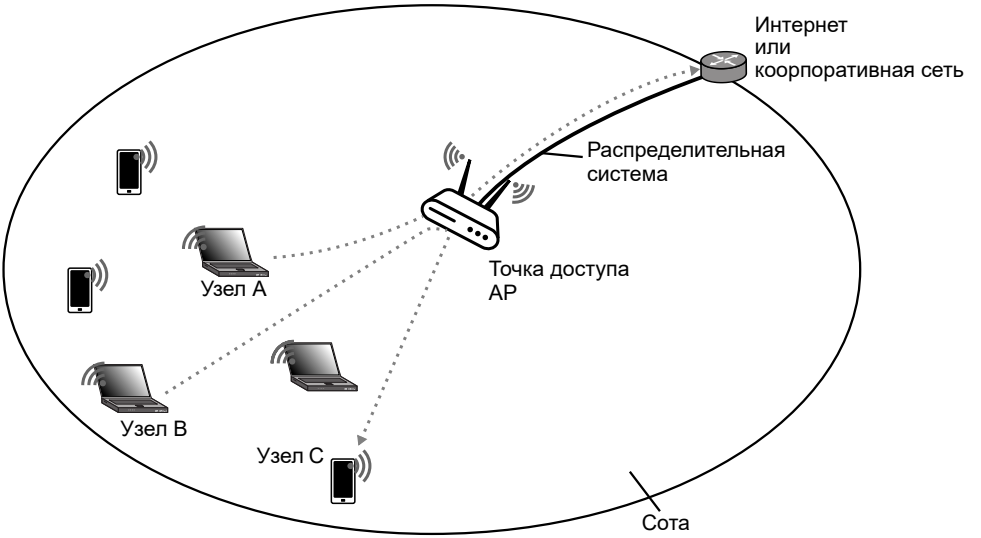


Рис. 25.1. Архитектура сети BSS

В этом примере точка доступа имеет проводное соединение с внешней по отношению к BSS сетью — это может быть сеть интернет-провайдера или корпоративная сеть. Такое соединение называется **распределительной системой** (Distributed System, **DS**). Оно не является обязательным элементом BSS, но в подавляющем большинстве применений сетей Wi-Fi оно присутствует, так как доступ к интернет-услугам или услугам корпоративной сети (и через нее — к услугам интернета) является одной из основных целей установки сетей Wi-Fi. Развитие интернета вещей может изменить эту ситуацию, когда все большее количество узлов домашней или офисной сети начнут обмениваться данными друг с другом, но пока такой обмен не очень интенсивен по сравнению с трафиком, идущим в интернет/корпоративную сеть.

Теоретически в такой сети возможны следующие варианты связей:

1) станция Wi-Fi — AP — внешний узел;
2) станция Wi-Fi — AP — станция Wi-Fi;
3) внешний узел — AP — станция Wi-Fi;
4) станция Wi-Fi — станция Wi-Fi.

Последний вариант связи, то есть непосредственный обмен данными между станциями, в сетях Wi-Fi не разрешается (за исключением ячеистых сетей Wi-Fi AdHoc, которые будет рассматриваться далее). Конечные узлы обмениваются кадрами между собой или

с внешними узлами *через точку доступа AP*. На рис. 25.1 показаны два варианта обмена данными в сети BSS.

В первом варианте узел *A* отправляет кадр в интернет. Этот процесс включает три этапа:

- получение узлом *A* доступа к разделяемой среде и побитная отправка кадра в радиосреду;
- точка доступа AP принимает все биты кадра в свой буфер;
- точка доступа AP отправляет кадр во внешнюю сеть.

Во втором варианте узел *B* отправляет кадр узлу *C*, являющемуся членом той же самой сети BSS. Процесс состоит из четырех этапов:

- получение узлом *B* доступа к разделяемой среде и побитная отправка кадра в радиосреду;
- точка доступа AP принимает все биты кадра в свой буфер;
- точка доступа AP получает доступ к среде и побитно передает кадр с адресом назначения узла *C* в разделяемую радиосреду;
- станция *C* принимает кадр в свой буфер.

Топология сети BSS

Описанные два варианта передачи кадров в сети BSS отражают логическую топологию этой сети. На рис. 25.2 показаны физические и логические связи сети, изображенной на рис. 25.1. Из рисунка видно, что из всего множества физических линий связи (на рисунке штриховые линии) сети BSS логические связи (сплошные линии) образованы только поверх тех физических линий, которые соединяют конечные узлы с точкой доступа, образуя **логическую топологию связей «звезда»**. Такая топология и определяет обмен данными в сети только через точку доступа, а непосредственные передачи кадров между конечными узлами этой топологией не поддерживаются.

Такая функциональность позволяет назвать точку доступа «повторителем», так как она повторяет принимаемые кадры, а можно назвать и «мостом», потому что кадры полностью буферизуются перед повторением. Оба эти определения используются при описании функциональности точки доступа сети Wi-Fi. При этом нужно понимать, что кавычки в обоих случаях необходимы, так как прямого соответствия с повторителями и мостами проводных сетей у точки доступа нет.

Возникает естественный вопрос: почему в архитектуре BSS не разрешается прямая передача кадров между конечными узлами, как, например, в локальной сети Ethernet, построенной на коммутаторе? Ведь такой способ передачи кадров между конечными узлами сети BSS нерационален — каждый кадр передается дважды, и дважды его передача монополизирует разделяемую среду, не давая ее использовать другим узлам сети.

Ответ состоит в том, что централизация функций упрощает организацию сети, а сосредоточение мощности и «интеллекта» сети в центральном узле позволяет конечным узлам быть проще, а значит, и дешевле, и потреблять меньше энергии, что очень важно для многих типов конечных узлов Wi-Fi. То есть разработчики технологии Wi-Fi приняли то же решение в пользу централизованного управления, что и разработчики технологии мобильных сотовых сетей.

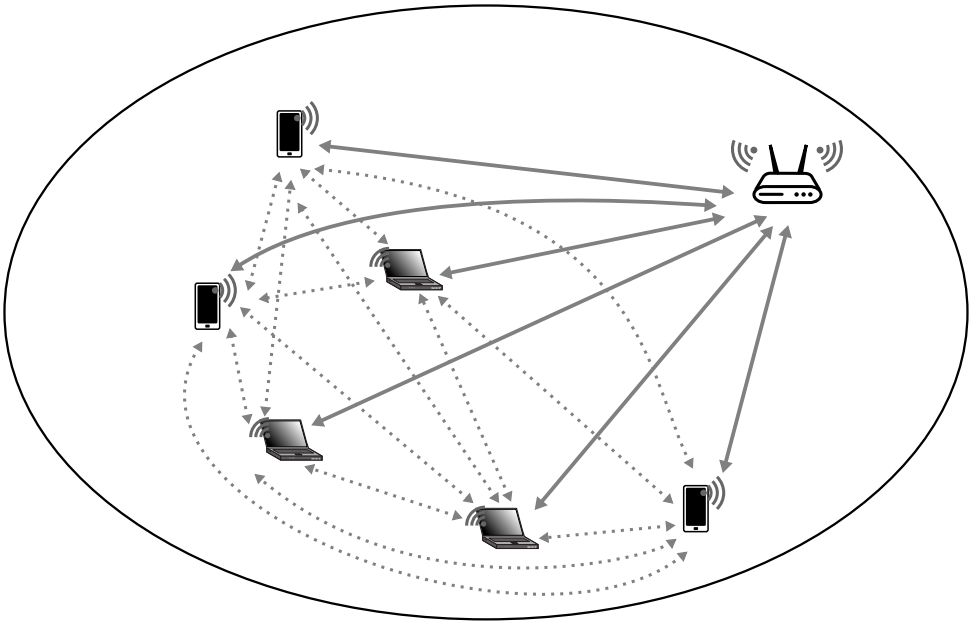


Рис. 25.2. Физические и логические связи сети BSS

Обычно радиопорт точки доступа имеет бóльшую мощность, чем конечный узел, и более высокую чувствительность. Это позволяет увеличить зону покрытия сети BSS даже в том случае, когда ее конечными узлами являются устройства с передатчиками небольшой мощности и невысокой чувствительности. Действительно, высокая мощность передатчика точки доступа (100–250 мВ и выше) позволяет смартфонам и ноутбукам принимать сигналы точки доступа Wi-Fi на достаточно больших расстояниях (у многих стандартов физической среды Wi-Fi — до 250 метров на открытом воздухе без препятствий), и в то же время высокая чувствительность антенны и приемника точки доступа позволяет устойчиво принимать на этом же расстоянии менее мощные сигналы своих клиентов.

Точка доступа в сети BSS осуществляет и другие важные для сети функции:

- Выполняет процедуру присоединения станции к сети. Для того чтобы получить возможность обмениваться данными в сети Wi-Fi, станция должна сначала присоединиться к сети. Логическое присоединение станции к сети называется **ассоциацией**.
- Осуществляет связь конечных узлов сети с интернетом (проводную или беспроводную). Для этого точка доступа включает в себя встроенный IP-маршрутизатор. В стандартах 802.11 эта функция точки доступа не описана, но на практике каждая точка доступа такой маршрутизатор имеет.
- Назначает IP-адреса конечным узлам с помощью своего DHCP-сервера.
- Может иметь проводные Ethernet-порты, для которых AP выполняет функции коммутатора (моста, теперь уже без кавычек). В таком случае разделяемая радиосреда является одним из сегментов этого моста. При передаче кадров между проводным и беспроводным сегментами AP выполняет функции транслирующего моста, так как форматы кадров Ethernet и Wi-Fi не совпадают.

Процедура присоединения к сети

Для того чтобы работать с точкой доступа, станция пользователя должна выполнить *процедуру присоединения к сети*. В ходе этой процедуры точка доступа может попросить станцию пройти аутентификацию, чтобы предотвратить доступ посторонних пользователей к данной сети.

Сеть BSS имеет два идентификатора:

- Символьное имя **SSID** (Service Set IDentifier) длиной до 32 символов дается сети владельцем или администратором локальной сети Wi-Fi. Оно обычно имеет понятный смысл — «Сеть аэропорта Быково» или «Семья Ивановых».
- Числовой идентификатор **BSSID** (Basic Service Set IDentifier) — это MAC-адрес AP. Он используется во всех процедурах взаимодействия станций с точкой доступа, однозначно идентифицируя точку доступа, например, при переходе пользователя из одной соты в другую.

Существуют два режима присоединения — *пассивный* и *активный*. В первом режиме станция пассивно прослушивает радиоэфир и пытается на всех частотных каналах обнаружить кадры специального типа — так называемые **маячковые кадры** (beacon frames). Эти кадры периодически, с интервалом 100 мс, посылает точка доступа. В них содержится ее MAC-адрес (BSSID) и идентификатор сети SSID. Активный режим отличается от пассивного тем, что станция сама посылает в радиосреду специальные кадры другого типа — **пробные кадры** (probe frames). Точка доступа, получив пробный кадр, отвечает на него, сообщая станции те же сведения о себе, что и в маячковых кадрах, то есть MAC-адрес и идентификатор SSID.

Станция может одновременно получать маячковые кадры от нескольких точек доступа. Обычно решение о присоединении к одной из доступных сетей принимает пользователь, когда он видит на экране список идентификаторов сетей SSID, к которым он может присоединиться, и выбирает одну из них. Станция может запомнить этот SSID и параметры аутентификации, использовавшиеся при присоединении к ней, в своем кэше, и в следующий раз попытаться присоединиться к этой сети автоматически, как только окажется в зоне покрытия точки доступа этой сети. Отметим, что пользователь имеет возможность запретить автоматическое подключение к некоторым сетям (посредством изменения ряда настроек устройства).

Когда сеть выбрана, станция посылает точке доступа кадр «Запрос присоединения». Затем точка доступа инициирует процедуру аутентификации, которая может быть двух типов — открытой (open authentication) или защищенной (secure authentication). При *открытой аутентификации* никакой аутентификации на самом деле не происходит, и любая станция, которая посылает запрос присоединения, считается успешно аутентифицированной. При *защищенной аутентификации* происходит проверка знания станцией заранее выбранного для данной сети пароля (выполняется строгая аутентификация, без передачи пароля по радиосреде) и в процессе этой проверки станция и точка доступа вырабатывают сессионные ключи, с помощью которых будет шифроваться трафик между станцией и точкой доступа. Подробнее процедура защищенной аутентификации рассмотрена в главе 29.

Если аутентификация успешна, то точка доступа отвечает кадром «Ответ присоединения», в котором сообщает станции, был ли запрос успешен, и при положительном ответе

точка доступа сообщает станции основные параметры работы сети: *уникальный временный идентификатор станции в сети AID* (12 бит, используется в некоторых процедурах вместо длинного MAC-адреса), выбранный диапазон частот, номер канала в этом диапазоне, а также некоторые другие параметры.

После успешной аутентификации маршрутизатор распределительной системы, пользуясь протоколом DHCP, назначает станции IP-адрес. Для адресов IPv4 маршрутизатор, скорее всего, выдаст частный IP-адрес, так что для доступа станции к интернету потребуется применение техники трансляции адресов NAT (см. главу 30). Для адресов IPv6 адрес, скорее всего, будет выдан глобальный, так что трансляция адресов не понадобится.

Стек протоколов

Стек протоколов сетей Wi-Fi имеет ту же структуру, что и стек протоколов Ethernet, и состоит из двух уровней:

- Физический уровень, на котором выполняется модуляция сигналов и побитная их передача в радиосреду в заданных частотных каналах.
- Канальный уровень, который включает в себя два подуровня:
  - подуровень доступа к разделяемой среде MAC, который может работать в двух режимах — децентрализованном, при котором все станции и точка доступа соревнуются за доступ к среде, или централизованном, при котором точка доступа распределяет ресурсы радиосреды между станциями;
  - подуровень LLC, выполняющий, как и в стеке Ethernet, функции согласования с протоколами верхних уровней, например с IP.

Подуровень MAC выполняет в сетях Wi-Fi следующие функции:

- Обеспечивает доступ к разделяемой среде.
- Организует надежный обмен кадрами данных канального уровня между узлами сети на основе распределенного или централизованного методов доступа.
- Предотвращает эффект скрытой станции.
- Обеспечивает поддержку QoS для разных классов трафика.
- Агрегирует кадры для более эффективного использования разделяемой среды.

Физический уровень BSS

Существует целый ряд спецификаций физического уровня сети Wi-Fi, которые отличаются друг от друга используемым частотным диапазоном, методом мультиплексирования и методом модуляции сигналов и, как следствие, скоростью передачи данных. Все варианты физического уровня работают с одним и тем же алгоритмом уровня MAC, но некоторые временны́е параметры уровня MAC зависят от используемого физического уровня.

По табл. 25.1, где представлены наиболее популярные варианты физического уровня стандартов семейства IEEE 802.11[1], можно проследить впечатляющий прогресс технологий

[1] Начиная с версии 802.11n, консорциум Wi-Fi дает своим стандартам номера: 802.11n = Wi-Fi 4 и т. д.

беспроводных локальных сетей Wi-Fi в течение последних почти трех десятилетий. К примеру, скорость передачи данных, поддерживаемая стандартом IEEE 802.11ax, который был ратифицирован в начале 2021 года, превышает скорость стандарта 802.11 1997 года в 10 тысяч раз!

Таблица 25.1. Стандарты семейства IEEE 802.11

Формальное и неформальное название	Рабочий диапазон частот	Максимальная скорость (без помех, на расстоянии 1 м между передатчиком и приемником)	Способ мультиплексирования	Год принятия
IEEE 802.11	2,4 ГГц	До 1,2 Мбит/с	DSSS/FHSS	1997
IEEE 802.1b	2,4 ГГц	До 11 Мбит/с	DSSS	1999
IEEE 802.11a	5 ГГц	54 Мбит/с	OFDM	1999
IEEE 802.11g	2,4 ГГц,	54 Мбит/с	OFDM, совместим с 802.11b DSSS	2003
IEEE 802.11n (Wi-Fi 4)	2,4/5 ГГц	288,8/ 600 Мбит/с Полоса 20 МГц Полоса 40 МГц	OFDM, 4 MIMO-потока	2009
IEEE 802.11ac (Wi-Fi 5)	5 ГГц	Полоса 20 МГц до 347 Мбит/с Полоса 40 МГц до 800 Мбит/с Полоса 80 МГц до 1733 Мбит/с Полоса 160 МГц до 3467 Мбит/с	OFDM 8 MIMO-потоков	2013
IEEE 802.11ax (Wi-Fi 6 и 6E)	2,4/5/6	До 9,6 Гбит/с	OFDMA MU-MIMO	2019 (ратификация в 2021)
IEEE 802.11be (Wi-Fi 7)	2,4/5/6	До 46 Гбит/с	OFDMA MU-MIMO	2024 (ратификация в 2026)

Как видно из табл. 25.1, узлы сети Wi-Fi могут работать в трех диапазонах частот: 2,4 ГГц, 5 ГГц и 6 ГГц, все они являются нелицензируемыми.

Диапазон 2,4 ГГц был первым диапазоном, отведенным для сетей Wi-Fi (по стандарту 802.11 1997 года). Этот диапазон шириной 83,5 МГц разбит на пользовательские каналы по 5 МГц (для взаимодействия пары узлов — пользователей радиосреды). В России разрешается использовать 13 каналов, со средней частотой первого канала 2,412 ГГц и средней частотой последнего, 13-го канала 2,427 ГГц.

Станция может использовать для передачи данных точке доступа полосу частот четырех объединенных каналов 5 МГц, то есть канал шириной в 20 МГц.

До появления технологии Wi-Fi диапазон 2,4 ГГц использовался различными научными, медицинскими и бытовыми приборами, которым не нужна была высокая скорость передачи данных. Этим объясняется его разбиение на достаточно узкие пользовательские каналы в 5 МГц. Такая ширина частотного канала не устраивала разработчиков технологии Wi-Fi, которые хотели добиться скоростей передачи данных по крайней мере в несколько мегабит в секунду. Поэтому каналы было решено объединить, так что вместо 13 каналов по 5 МГц получилось 3 канала шириной 22 МГц (рис. 25.3). Объединенные каналы иден-

тифицируются номерами своих центральных подканалов 1, 6 и 11. Именно они являются разрешенными для работы узлов сети Wi-Fi.

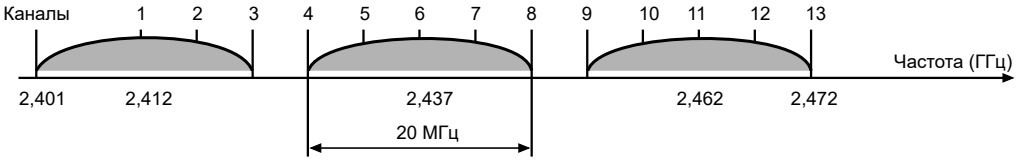


Рис. 25.3. Каналы диапазона 2,4 ГГц

Диапазон 5 ГГц выделен почти в полное распоряжение сетей Wi-Fi. «Почти» означает, что сетям Wi-Fi приходится делить его с радарами, которые начали его использовать ранее.

В этом диапазоне для сетей Wi-Fi отводится 24 неперекрывающихся частотных канала шириной 20 МГц, при этом более поздние стандарты, начиная с Wi-Fi 4 (802.11n), позволяют объединять соседние каналы в более крупные каналы 40, 80 и 160 МГц.

Диапазон 6 ГГц отведен сетям Wi-Fi в исключительное использование и занимает частоты от 5,925 ГГц до 7,125 ГГц, это 59 каналов по 20 МГц. Такая широкая полоса частот дает возможность резко увеличить предельные скорости передачи данных и разворачивать новые сети Wi-Fi в тех местах, где частоты диапазонов 2,4 и 5 ГГц уже заняты или испытывают воздействие интенсивных помех. В этом диапазоне работают сети последних двух стандартов — Wi-Fi 6E (802.11ax) и Wi-Fi 7 (802.11 be).

Развитие стандартов 802.11 шло также в направлении усовершенствования методов мультиплексирования потоков данных различных пользователей. Мультиплексирование OFDM повысило спектральную эффективность технологии Wi-Fi, а применение мультиплексирования OFDMA наряду с техникой MU-MIMO в последних версиях стандарта позволило более эффективно обслуживать соты Wi-Fi с большим количеством пользователей.

В отличие от Ethernet, в технологии Wi-Fi помимо кадров канального уровня (MAC-уровня) существуют кадры физического уровня. Кадр физического уровня носит название **PPDU** (Physical Layer Protocol Data Unit). Заголовок PPDU, помимо преамбулы, предназначенной для вхождения приемника в побитовую и покадровую синхронизацию с передатчиком, несет в себе информацию о типе модуляции, примененном к полю данных физического кадра (а в этом поле данных находится кадр канального уровня), о структуре частотных каналов (на какие подканалы разбит канал), а в последних версиях стандарта 802.11, в которых применяется мультиплексирование OFDMA, еще и информацию о распределении ресурсов радиосреды между станциями.

MAC-уровень

Формат кадров MAC-уровня

Кадры MAC-уровня технологии Wi-Fi отличаются разнообразием типов (еще одно отличие от Ethernet, вызванное спецификой радиосреды) и имеют сложную структуру, показанную на рис. 25.4.



Рис. 25.4. Формат кадра Wi-Fi

Имеется три типа кадров Wi-Fi:

- кадры данных;
- кадры слоя управления;
- кадры слоя менеджмента.

Тип кадра, а также его подтип определяются значением соответствующих полей двухбайтного поля *«Управление кадром»*.

Одной из особенностей кадра данных Wi-Fi является наличие в нем *четырех полей MAC-адресов*. Их назначение зависит от конфигурации сети. В том случае, когда две станции находятся в различных BSS и они взаимодействуют через две точки доступа, *соединенные радиосредой*, используются все четыре MAC-адреса: станции источника (Address 4), двух точек доступа (Address 1 и Address 2) и станции назначения (Address 3). В остальных случаях используется только три адреса — станции источника, точки доступа и станции назначения (или же маршрутизатора распределительной системы).

> Три или четыре адреса нужны узлам сети Wi-Fi из-за обязательной промежуточной передачи кадров через точку доступа, которую нужно адресовать.

Кадр Wi-Fi имеет ряд полей, помогающих узлам, обменивающимся данными, обнаруживать и восстанавливать искаженные и потерянные кадры. Искажение данных выявляется по значению поля контрольной суммы, а потерянные кадры — по значению поля *«Управление последовательностью»*, содержащем порядковые номера кадров.

«Поле данных» позволяет переносить до 2312 байт пользовательских данных. Кадры Wi-Fi могут фрагментироваться, при этом назначение фрагментирования здесь отличается от назначения аналогичной операции в протоколе IP. Там пакет фрагментируется в том случае, когда его размер превосходит максимальную величину транспортного блока (MTU) в промежуточном маршрутизаторе. Здесь же фрагментация применяется для *ускорения* передачи данных в условиях высокого уровня помех в радиосети, так как чем меньше размер кадра, тем выше вероятность, что он будет получен неискаженным. Станция назначения подтверждает квитанцией каждый фрагмент кадра, а станция-отправитель не передает следующий фрагмент, пока не получит квитанции. Таким образом, при искажении битов в радиосреде станция-отправитель передает повторно не весь кадр, а только его фрагмент. Для поддержки фрагментирования в поле «Управление последовательностью» указываются не только номер кадра, но и номер фрагмента. Однобитовый признак *«Фрагмент»* всех фрагментов, кроме последнего, равен 1 (значение 0 говорит о том, что это последний фрагмент).

Поле *«Длительность»* определяет время занятости среды, которое станция дополнительно резервирует за собой после передачи данного кадра. Это время определяется режимом доступа к среде и типом кадра, передаваемого станцией.

Единица в поле *«Шифрование»* означает, что пользовательские данные зашифрованы.

Единица в поле *«Еще данные»* означает, что у станции имеются данные, которые она хочет передать в следующих кадрах.

Единица в поле *«Порядок»* уведомляет принимающую станцию, что порядок кадров имеет значение.

Поля *«К точке доступа»* и *«От точки доступа»* показывают направление передачи кадра и определяют назначение полей адресов следующим образом:

Когда станция STA1 хочет послать кадр станции STA2, то сначала она посылает этот кадр точке доступа AP, при этом бит «К точке доступа» равен 1, бит «От точки доступа» равен 0 и поля адресов имеют следующие значения:

- Адрес 1 = MAC(AP) — этот адрес всегда имеет значение адреса следующего хопа.
- Адрес 2 = MAC(STA1) — этот адрес всегда имеет значение адреса *непосредственного* отправителя, им была станция STA1.
- Адрес 3 = MAC(STA2) — этот адрес имеет значение адреса финального назначения, то есть станции STA2.
- Адрес 4 в этой передаче не используется.

Когда точка доступа AP пересылает этот кадр по назначению, к станции STA2, то бит «К точке доступа» равен 0, бит «От точки доступа» равен 1, а поля адресов выглядят следующим образом:

- Адрес 1 = MAC(STA2) — этот адрес всегда имеет значение адреса следующего хопа.
- Адрес 2 = MAC(AP) — этот адрес всегда имеет значение адреса *непосредственного* отправителя, им была точка доступа.
- Адрес 3 = MAC(STA1) — в том случае, когда кадр направлен от точки доступа, этот адрес имеет значение адреса источника, то есть станции STA1.
- Адрес 4 в этой передаче не используется.

Четвертый адрес используется только при беспроводной передаче кадров между станциями, находящимися в различных BSS. Мы рассмотрим этот случай позже.

В поле *«Управление кадром»* кадра Wi-Fi имеется бит *«Управление питанием»*, который станция устанавливает в 1 в любое время, которое ей кажется подходящим для перехода в спящий режим, когда она не передает и не принимает данные, тем самым экономя энергию батарей. Точка доступа, получив сигнал от станции, перестает посылать ей кадры данных, и если они поступают от внешней сети в течение неактивности станции, то точка доступа их буферизует. Правда, «сон» станции очень короток: она ставит себе таймер-будильник на 100 мс, чтобы проснуться к следующему кадру-маячку. Проснувшись по таймеру, станция дожидается прихода очередного кадра-маячка от точки доступа. В этом кадре точка доступа передает список тех станций, для которых у нее есть буферизованные кадры. Найдя себя в списке, станция может служебным кадром запросить передачу буферизованных кадров. Если же за время периода неактивности кадры для станции не поступали, то станция может снова перейти в спящий режим до прихода следующего кадра-маячка. Таким образом, в тех случаях, когда у станции нет собственных кадров для передачи и никто ей не посылает кадры, она может проводить почти все время в «спячке». Энергию батарей ноутбуков и телефонов желательно экономить, и в стандарте Wi-Fi содержится основа для реализации процедур энергосбережения.

Доступ к разделяемой среде

В стандартах 802.11 описываются два метода доступа к разделяемой среде:

- Первый, являющийся обязательным для реализации, имеет два равнозначных названия: **распределенный режим доступа** к разделяемой среде (Distributed Coordination Function, **DCF**) и **CSMA/CA** (Carrier Sense Multiple Access with Collision Avoidance) — метод прослушивания несущей частоты с множественным доступом и предотвращением коллизий. Далее в основном будет использоваться название CSMA/CA.
- Второй, метод **централизованного управления доступом**, сочетающий централизованное распределение разделяемой среды между станциями с распределенным методом доступа CSMA/CA, стал применяться начиная со стандарта Wi-Fi 6 (IEEE 802.11ax).

Методы доступа не зависят от типа сети Wi-Fi. Они работают одинаково как в сетях BSS, так и в сетях других типов — ESS и ячеистой сети, которые изучаются позже.

Метод доступа CSMA/CA близок по своей идее к методу доступа CSCM/CD, применявшемуся в ранних сетях Ethernet с разделяемой средой (см. главу 10), но отличается от него тем, что вместо *допущения коллизий* и оперативного реагирования на них он пытается коллизии *предотвратить*.

Метод доступа CSMA/CD работает в беспроводных локальных сетях неэффективно по двум причинам:

- Сложно создать приемопередатчик, который одновременно передает и прослушивает среду.
- Он не учитывает эффект скрытой станции. Такая ситуация вполне возможна в BSS с точкой доступа, когда обе станции, пытающиеся получить доступ к среде, находятся в зоне покрытия точки доступа, но не в зоне покрытия друг друга.

В методе CSMA/CA коллизии определяются косвенно. Для этого каждый переданный кадр должен подтверждаться **кадром положительной квитанции ACK**, посылаемым станцией-получателем. Если же по истечении оговоренного интервала времени квитанция не поступает, то станция-отправитель считает, что произошла коллизия. Такой же метод обнаружения коллизий использовался в первых радиосетях ALOHA. Квитанция ACK всегда посылается непосредственным получателем кадра, то есть если станция STA1 посылает кадр станции STA2 в той же соте BSS через точку доступа, то точка доступа, получив этот кадр, отправляет квитанцию ACK станции STA1, а после того, как этот кадр от точки доступа получит станция STA2, она отправляет квитанцию ACK точке доступа.

Между получением последнего бита кадра и отправкой квитанции ACK выдерживается пауза — так называемый **короткий межкадровый интервал** (Short Inter Frame Space, **SIFS**). Значение интервала SIFS выбирается таким, чтобы принимающая станция успела обработать принятый кадр и послать ответ. В методе доступа CSMA/CA, как мы увидим дальше, используется несколько типов межкадровых интервалов.

Использование квитанций ACK является механизмом реакции сети на коллизию, но не ее предотвращения. Для предотвращения коллизий применяются методы, в которых станции следят за состоянием разделяемой среды и выбирают момент для передачи своего кадра таким образом, чтобы вероятность коллизии была как можно меньше.

Каждая станция, поддерживающая CSMA/CA, должна прослушивать среду и уметь определять, занята она или нет, — как и в Ethernet на разделяемой среде. При обнаружении передачи кадра каждая станция должна его буферизовать и сравнить адрес назначения со своим собственным. Если адреса совпадают, то станция проверяет контрольную сумму кадра и, если она корректна, передает кадр своим протоколам верхних уровней.

Если же адрес назначения не совпадает с ее собственным адресом, то станция читает значение поля «Длительность» в заголовке данного кадра и запоминает его в своем внутреннем буфере, называемом **вектором распределения сети**, или **таймером NAV** (Network Allocation Vector).

Значение поля «Длительность» устанавливается передающей станцией. Оно говорит всем станциям соты, чтобы они в течение этого времени не пытались получить доступ к разделяемой среде, так как среда будет занята передачей ее кадра. «Длительность» является расчетной величиной, равной сумме трех составляющих:

- длительности передачи кадра;
- длительности межкадрового интервала SIFS;
- длительности передачи положительной квитанции ACK от принявшей кадр станции.

Таким образом, операция передачи кадра и получения подтверждения становится *неделимой транзакцией* — никакая другая станция не может вклиниться в эту операцию, пока она не завершится либо получением подтверждения, либо истечением тайм-аута ожидания подтверждения. При передаче кадра данных с широковещательным адресом время длительности владения средой устанавливается в 0, так как такой кадр не требует подтверждения.

Метод доступа требует синхронизации станций, чтобы они не начинали передавать данные в одно и то же время. Точками синхронизации являются моменты окончания передачи очередного кадра (рис. 25.5). Никакие специальные синхронизирующие сигналы не используются.

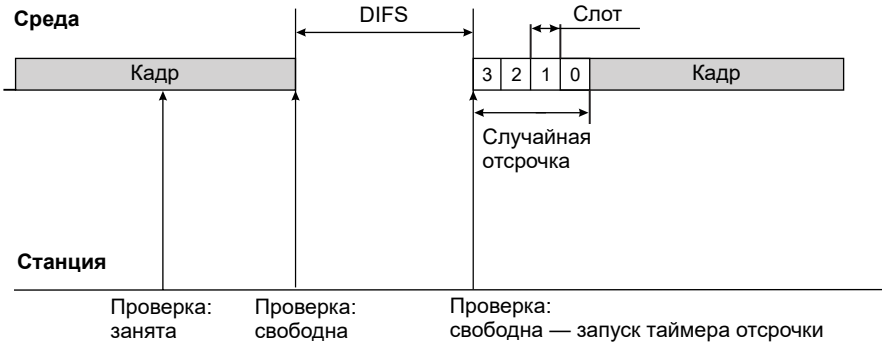


Рис. 25.5. Метод доступа CSMA/CA

Предотвращение коллизий достигается за счет того, что каждая станция выбирает для передачи своего кадра случайный момент времени, отсчитываемый от момента окончания передачи последнего по времени кадра. Вероятность того, что несколько станций выберут один и тот же случайный момент, существует — и тогда произойдет коллизия, но она значительно меньше, чем вероятность коллизии при использовании метода CSMA/CD, когда станции разрешается передавать кадр сразу же после окончания межкадрового интервала.

Случайный интервал состоит из двух составляющих:

- **распределенного межкадрового интервала** (Distributed InterFrame Space, **DIFS**) и
- **случайной отсрочки**.

Значение межкадрового интервала DIFS равно постоянной закрепленной в стандарте величине, такой, что при определенном методе кодирования и мультиплексирования данных в радиосреде все станции сети могут зафиксировать окончание передачи кадра, независимо от своего положения в зоне покрытия сети, потому что за это время сигнал успеет дойти до самой дальней станции. В современных сетях Wi-Fi наряду с фиксированным интервалом DIFS для этих же целей применяется набор адаптивных межкадровых интервалов (AC_VO, AC_VI, AC_BE, AC_BK), используемых для обеспечения качества облуживания для различных типов трафика. Мы рассмотрим этот механизм далее.

Случайная отсрочка отсчитывается с момента истечения межкадрового интервала DIFS. Она равна целому числу **тайм-слотов**. Длительность тайм-слота задана стандартом. Количество слотов, после истечения которых станция может передать кадр, выбирается как случайное целое число N, равномерно распределенное в интервале [0, CW], называемом **конкурентным окном** (Contention Window, **CW**). Чем больше это окно, тем менее вероятен выбор несколькими станциями одного и того же слота для передачи кадра. Случайное число N записывается в счетчик, называемый **таймером отсрочки**. Станция прослушивает среду после истечения каждого слота, и каждый раз, если среда оказывается свободной, станция вычитает из своего таймера отсрочки единицу. Если таймер отсрочки стал равен нулю, то есть время случайной отсрочки исчерпано, станция начинает передавать свой кадр.

Если же при очередной проверке слота среда оказалась занятой, вычитания единицы из таймера отсрочки не происходит. С этого момента станция начинает новый цикл борьбы за доступ к среде. Однако в этом новом цикле учитывается предыстория: вместо того, чтобы вычислять, а затем использовать новое значение случайной отсрочки, станция использует *остаток* от предыдущей, недоиспользованной случайной отсрочки. То есть если в предыдущем цикле борьбы за доступ к среде значение таймера случайной отсрочки было N, а в момент захвата среды другой станцией таймер уменьшился до значения K, то значение K и будет новым начальным значением таймера для нового цикла. В примере на рис. 25.5 начальное значение таймера отсрочки оказалось равным 3. При прослушивании слотов каждый раз среда оказывалась свободной, поэтому станция досчитала до 0 и начала передавать свой кадр.

Если коллизия все же случилась, то, не получив вовремя квитанции, станция пытается передать кадр повторно. При каждой повторной неудачной попытке передачи кадра значение конкурентного окна удваивается. А значит, вероятность повторной коллизии уменьшается. Как и в методе CSMA/CD, в данном методе количество неудачных попыток передачи одного кадра ограниченно, но стандарт 802.11 не дает точного значения этого верхнего предела. Когда верхний предел в N попыток достигнут, кадр отбрасывается, а счетчик последовательных коллизий устанавливается в ноль.

Предотвращение эффекта скрытой станции

Напомним, что эффект скрытой станции возникает в том случае, когда две станции соты (обозначим их A и B) не находятся в зоне покрытия друг друга, но обе находятся в зоне покрытия точки доступа и точка доступа находится в их зонах покрытия. Поэтому, если одна из этих станций начинает передачу кадра точке доступа, то другая не замечает этой передачи и, считая, что среда соты свободна, может начать сама передавать свой кадр и тем самым вызовет коллизию. Для предотвращения такой ситуации в методе доступа CSMA/CA предусмотрена процедура, основанная на служебных кадрах **RTS** (Request To Send — запрос на передачу) и **CTS** (Clear To Send — свободна для передачи). Она включает следующие этапы (рис. 25.6):

1. *Станция A проверяет канал, который кажется свободным* (станция B скрыта, поэтому станция A не может ее слышать). Станция A обычным образом ожидает в течение интервала DIFS, а затем в течение случайной задержки. Если среда была свободной все это время, станция A получает доступ к среде передачи.
2. *Станция A передает кадр RTS точке доступа.* Чтобы зарезервировать среду передачи за собой, станция A помещает в поле «Длительность» кадра RTS значение, равное сумме времен передачи кадров CTS, DATA, ACK и трем интервалам SIFS. Как видно из рис. 25.6, это соответствует предстоящей длительности передачи, если точка доступа подтвердит свою готовность принять данные. Все станции, кроме скрытой станции B, получают эту информацию и замолкают.
3. *Точка доступа корректно принимает RTS и отсылает кадр CTS.* Чтобы иметь возможность послать ответный кадр CTS, точка доступа захватывает среду «неконвенциональным» методом: вместо положенного интервала DIFS перед отправкой кадра она ожидает в течение короткого интервала SIFS. Поскольку SIFS меньше, чем DIFS, никакая другая станция не может помешать отправке кадра CTS. В кадре CTS обновлено поле «Длительность».

4. *Скрытая станция B слышит CTS.* Хотя скрытая станция *B* не слышала RTS, посланный станцией *A*, она слышит CTS от точки доступа и откладывает передачу данных, даже если среда кажется ей свободной. *В этот момент опасность коллизии предотвращена.*
5. Все станции, которые принимают RTS или CTS, узнают из них значение поля «Длительность», устанавливают свои таймеры NAV, замораживают счетчики задержки и замолкают. Конкуренция отсутствует.
6. После интервала SIFS станция *A* отправляет кадр данных.
7. Точка доступа отправляет подтверждение ACK станции *A* и ожидает SIFS.
8. Таймеры NAV истекают на всех остальных станциях, среда становится свободной, возобновляется обычная конкуренция.

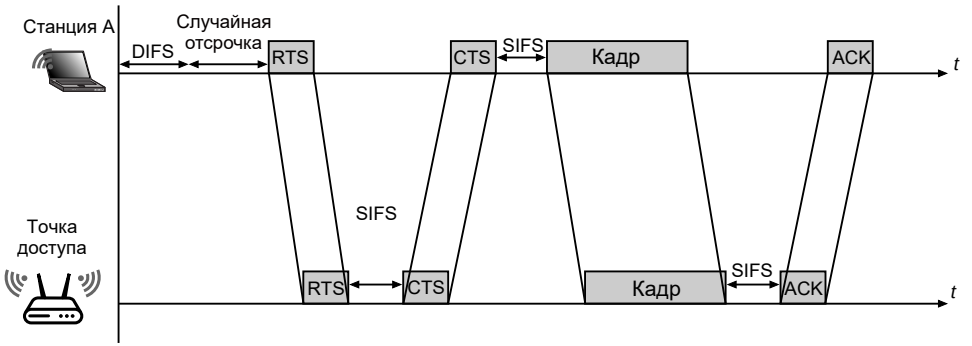


Рис. 25.6. Цикл передачи кадра

Механизм RTS/CTS не является обязательным и может быть включен или выключен администратором сети. Обычно в небольших сетях, где вероятность появления скрытой станции мала, его выключают. Но даже если этот механизм и включен, то станция его использует только тогда, когда размер кадра, который она собирается отправить, превышает некоторый порог, называемый «порогом RTS». То есть этот механизм используется только для больших кадров, в течение передачи которых вероятность коллизии из-за наличия скрытого терминала велика. Считается, что небольшие кадры более рационально отправлять без предварительной проверки скрытого терминала кадрами RTS/CTS: выигрыш от предотвращения редкой коллизии не превышает накладных расходов — дополнительного периода занятости среды этими кадрами.

Возникает вопрос: каким образом обеспечивается неделимость транзакции, состоящей в цикле RTS-SIFS-CTS-SIFS-DATA_FRAME-SIFS-ACK, изображенном на рис. 25.6?

По логике алгоритма доступа станция, выигравшая соревнование за доступ к среде, захватывает ее в монопольное владение на все время передачи своего кадра. Однако здесь среда доступа делится между станцией-отправителем и станцией-получателем, кадры которых чередуются. И тогда почему другие станции не могут вклиниться в эту довольно длинную последовательность кадров? Ответ заключается в том, что в соответствии с принятым алгоритмом доступа станциям — участникам обмена разрешается делать между кадрами этого цикла *короткие межкадровые интервалы* SIFS. Так как пауза DIFS длиннее, чем SIFS, то другие станции, не участвующие в обмене, не успевают захватить среду — они вынуждены ждать, пока весь цикл обмена закончится.

Поддержка QoS

Механизм доступа к среде CSMA/CA был усовершенствован консорциумом Wi-Fi Alliance в 2003 году для поддержки QoS. Этот механизм получил название Wi-Fi MultiMedia (**WMM**). Затем он вошел в стандарт 802.11e как усовершенствованный режим DCF (Enhanced DCF, **EDCF**).

Согласно стандарту EDCF трафик делится на 4 категории доступа (Access Categories, **AC**):

- AC_VO — высший приоритет, трафик голосовых сервисов.
- AC_VI — высокий приоритет, трафик видеосервисов.
- AC_BE — стандартный приоритет, трафик для обслуживания «по возможности» (Best Effort).
- AC_BK — низший приоритет, фоновый трафик (BacKground).

Значение категории трафика переносится в дополнительном поле заголовка кадра Wi-Fi «Управляющее слово QoS».

Приоритетное обслуживание трафика определенного класса достигается за счет изменения параметров алгоритма распределенного доступа — величин межкадрового интервала и конкурентного окна. Очевидно, что чем приоритетнее трафик, тем меньше для него должны быть эти значения, чтобы его кадры имели бóльшую вероятность захвата среды. Как видно из рис. 25.7, на котором интервалам присвоены имена соответствующих классов трафика, у трафика AC_VO эти значения минимальные, а у трафика AC_BK — максимальные, как и должно быть в соответствии с требованиями QoS. Конкретные значения интервалов AC зависят от стандарта физического уровня, используемого сетью. Станции, которые не поддерживают механизм EDCF, по-прежнему используют интервал DIFS.

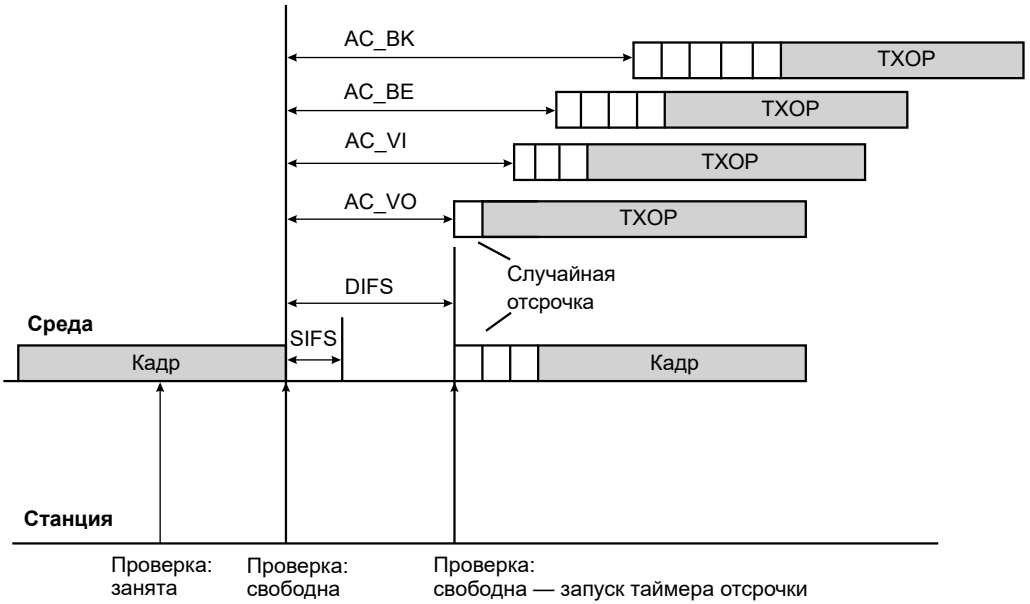


Рис. 25.7. Адаптивные межкадровые интервалы

Для каждой категории трафика определяется так называемое **время возможной передачи данных** (Transmitting Opportunity, **TXOP**) — это непрерывный интервал времени, в течение которого станции разрешается передавать свои кадры непрерывно, без повторного участия в конкурентной борьбе за доступ к разделяемой среде. Обычно для двух низших категорий трафика интервал TXOP равен нулю. Это условное значение, которое означает, что после окончания интервала случайной отсрочки станция может передать свой кадр, но только один, как это и предусмотрено алгоритмом CSMA/CA. После передачи одного кадра данных станция должна снова участвовать в соревновании за доступ к среде. Для остальных категорий трафика станция, получив доступ к среде, передает столько кадров, сколько позволяет время TXOP.

Агрегирование кадров

Методы предотвращения коллизий, основанные на выдерживании случайных пауз перед передачей кадра и на проверке доступности среды с помощью зондирующих команд RTS/CTS, а также проверки возникновения коллизии с помощью квитанций ACK приводят к большим накладным расходам при передаче кадров данных. Статистические данные показывают[1], что только в течение 15 % времени радиосреда передает данные пользователей, а все остальное время тратится на служебные кадры, периоды ожидания доступа к среде и межкадровые интервалы. Поэтому проблема повышения эффективности использования среды в сетях Wi-Fi стоит остро. Одним из средств повышения этой эффективности является механизм агрегирования данных и кадров, активно применяемый начиная с версии 802.11n.

Механизм агрегирования данных и кадров предусматривает агрегирование двух типов:

- **A-MSDU** — агрегирование данных пакетов верхних уровней в одном поле данных одного кадра;
- **A-MPDU** — агрегирование нескольких кадров в составной кадр.

При агрегировании A-MSDU несколько пакетов верхнего уровня, например пакетов IP, помещаются в поле данных одного кадра. Все пакеты должны быть адресованы одному и тому же получателю (STA или AP) и должны быть обслужены в соответствии с одним и тем же приоритетом QoS. Кадр A-MSDU имеет обычную структуру — один заголовок и одну контрольную сумму. О том, что внутри кадра находятся несколько пакетов верхнего уровня, получатель узнает из значения специального бита в заголовке кадра. Максимальный размер кадра A-MSDU — 7935 байтов (в поздних стандартах 11 383). Выигрыш от агрегирования здесь очевиден: исключается повторение всех служебных кадров RTS/CTS, ACK и пауз SIFS для каждого из пакетов. На рис. 25.8, *а* показано два цикла передачи двух кадров. У каждого имеется свой заголовок (H1, H2), поле данных (D1, D2) и контрольная сумма (CRC1, CRC2).

На рис. 25.8, *б* показана передача агрегированного кадра. Сокращение времени передачи при агрегировании A-MSDU по сравнению со стандартной передачей двух отдельных кадров достигается за счет того, что, во-первых, уменьшается число служебных кадров и межкадровых интервалов, а во-вторых, вместо двух заголовков суммарные данные сопровождаются одним заголовком и одной контрольной суммой.

---

[1] David Coleman. Wi-Fi 6 & 6E For Dummies®, Extreme Networks Special Edition, John Wiley & Sons, Inc. 2023.

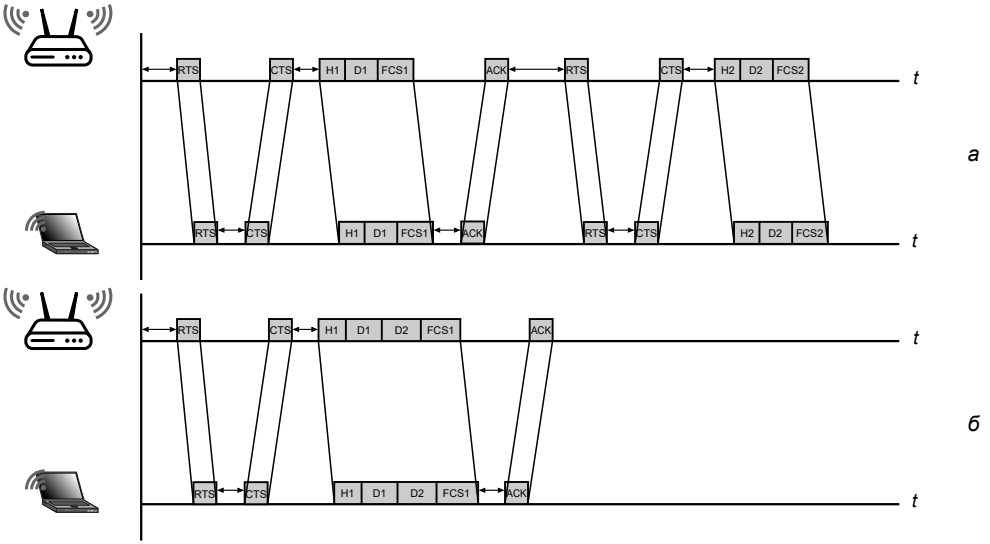


Рис. 25.8. Агрегирование кадров MSDU

Недостатком этого метода является необходимость отправки составного кадра по одному и тому же адресу (он не позволяет агрегировать кадры, предназначенные для разных адресатов), а это важно для передачи нескольких кадров от точки доступа к различным станциям — адресатам.

Кроме того, при искажении информации в любом пакете приходится повторно передавать весь кадр, поскольку в этом методе нет механизма выборочного повторения.

Агрегирование A-MPDU свободно от этих недостатков. Оно позволяет передать в радиосреду несколько кадров различным адресатам, объединенным в последовательность кадров, воспринимаемую методом доступа как один кадр. В такой последовательности каждый кадр оформлен стандартным способом, то есть имеет собственный заголовок с адресом назначения и контрольную сумму. Алгоритмом доступа к среде такая последовательность кадров рассматривается как один кадр, так как объединение кадров происходит не на MAC-уровне, а на физическом уровне и объединенный кадр является одним кадром PPDU со своей преамбулой и заголовком (рис. 25.9). В заголовке PHY объединенного кадра указывается, что этот кадр является кадром A-MPDU, а между отдельными кадрами на физическом уровне вставляется специальная последовательность бит — разделитель кадров (на рисунке не показан).

При обнаружении кадра A-MPDU в радиосреде все станции начинают его буферизовать и анализировать поле адреса назначения в каждом из составляющих его кадров. Если станция обнаруживает свой адрес, то она считывает этот кадр и обрабатывает его стандартным образом, при этом продолжая сканировать последовательность кадров в составном кадре A-MPDU, выбирая оттуда другие предназначенные ей кадры.

Как мы видим, процесс получения кадров из агрегированного кадра A-MPDU достаточно прост. Сложнее организовать подтверждение приема кадров различными станциями с помощью квитанций ACK. Каждая станция, которая получила один или несколько кадров из агрегированного кадра, должна отправить квитанцию Block ACK. Эта квитанция

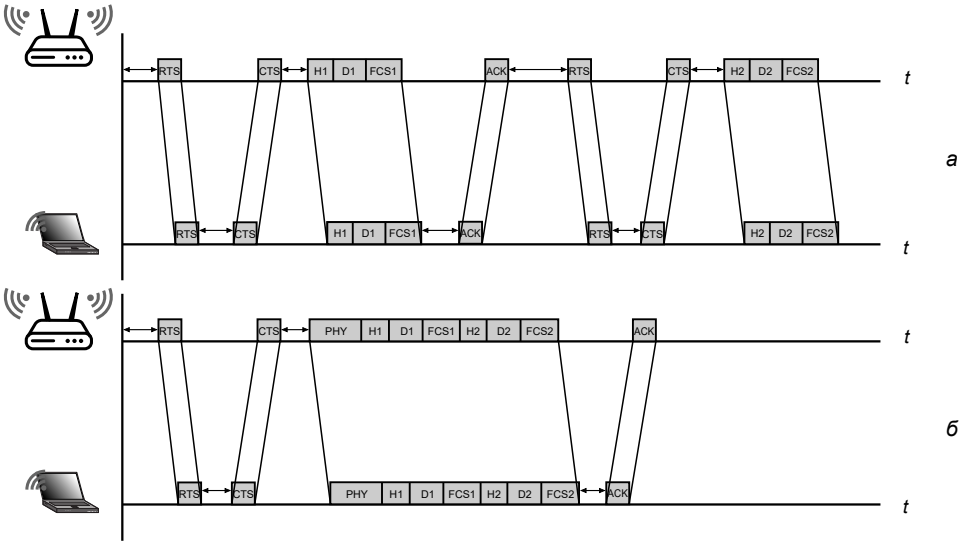


Рис. 25.9. Агрегирование кадров MPDU

отличается от стандартной квитанции ACK тем, что содержит битовую карту подтверждений приема кадров — по одному биту на каждый кадр. Например, если в агрегированном кадре было 8 кадров и станции STA3 предназначалось три из них, то битовая карта 110 означает, что первые два кадра были получены с корректной контрольной суммой, а третий — нет и его необходимо повторить. Необходимость повторной передачи всего агрегированного кадра при искажении информации в одном из кадров отпадает.

Но это еще не полное решение проблемы. Если разрешить каждой станции, получившей кадр, отсылать квитанцию Block ACK независимо от других станций, то квитанции могут вызвать коллизию и ситуация станет неопределенной.

Поэтому процесс отправки квитанций упорядочивается: точка доступа отправляет служебное сообщение Block Ack Request (BAR), то есть запрос на отправку квитанции, каждой станции по очереди, так что ответы Block ACK от разных станций столкнуться в радиосреде не могут. Но это, конечно, удлиняет процесс передачи агрегированного кадра.

Максимальная длина кадра A-MPDU составляет 65 535 байт в версии стандарта 802.11n, а в последних версиях она увеличена до 1 Мбайт при максимальном количестве отдельных кадров внутри агрегированного кадра — 1024.

Централизованный доступ к среде

Начиная со стандарта Wi-Fi 6 (IEEE 802.11ax), в сети Wi-Fi на смену технике мультиплексирования OFDM пришла **техника мультиплексирования OFDMA**. Мы рассмотрели основы этой техники в главе 23, а ее конкретную реализацию в мобильных сотовых сетях — в главе 24.

Вместе с мультиплексированием OFDMA в сети Wi-Fi пришел новый метод доступа к разделяемой радиосреде — **метод централизованного управления доступом**, при котором

центральный узел (в данном случае точка доступа) определяет расписание доступа станций соты к среде и сообщает его станциям в служебных кадрах, а станции просто выполняют его, не проявляя никакой инициативы.

Для совместимости с оборудованием предыдущих стандартов метод централизованного доступа дает возможность станциям, его не поддерживающим, бороться за доступ к разделяемой среде на основании алгоритма CSMA/CA. Точка доступа также вступает в это соревнование и, когда получает доступ к среде, разделяет среду между станциями сети уже централизованно, на период доставшегося ей по алгоритму CSMA/CA времени TXOP.

Сначала коротко рассмотрим специфику OFDMA в стандарте Wi-Fi 6 на примере мультиплексирования данных в канале шириной 20 МГц (в каналах другой ширины все основные свойства Wi-Fi OFDMA сохраняются). Канал шириной 20 МГц делится на подканалы шириной в 78,125 кГц. Подканал получил название **тон**. Таким образом, в канале 20 МГц имеется 256 тонов. (Для сравнения: в стандартах с мультиплексированием OFDM расстояние между подканалами равно 312,5 кГц и число подканалов в канале 20 МГц равно 64.)

Тоны могут объединяться в **ресурсные единицы** (Resource Unit, **RU**, — аналог ресурсных блоков LTE). В отличие от сетей LTE ресурсная единица Wi-Fi может иметь различную ширину: 26, 52, 106 и 242 тона. При этом в каждом цикле планирования могут использоваться ресурсные единицы разного размера, например, станциям могут быть выделены четыре ресурсные единицы в 26 тонов и одна в 106.

Еще одним отличием Wi-Fi OFDMA от LTE/5G OFDMA является отсутствие фиксированного размера ресурсной единицы во времени. То есть, в отличие от ресурсного блока LTE, который во временной оси состоит из 7 символов (или 14 символов в NR 5G), ресурсная единица Wi-Fi OFDMA имеет переменную длительность — она определяется длительностью кадра, передаваемого в этой ресурсной единице. А так как в различных ресурсных единицах передаются в общем случае кадры различной длины, то весь цикл передачи в рамках периода TXOP имеет длительность самого продолжительного кадра в наборе передаваемых кадров.

Передача данных происходит в режиме дуплексного разделения времени TDD, то есть в направлениях DL и UL используется один и тот же частотный канал 20/40/80/160 МГц.

Важной проблемой для сетей Wi-Fi на основе OFDMA является **обеспечение совместимости** со станциями, работающими по стандартам предыдущих поколений. Для этого все служебные кадры, которые должны быть понятны унаследованным станциям, посылаются в технике OFDM, в частотных подканалах шириной 312,5 кГц, которые всегда использовались в ранних версиях Wi-Fi.

Теперь рассмотрим **параллельную передачу кадров в нисходящем направлении DL** (рис. 25.10):

1. Планировщик MAC уровня точки доступа принимает решение по распределению ресурсных единиц RU между станциями (мы предполагаем, что в буферах точки доступа имеются кадры, принятые из интернета или от других станций соты, адресованные некоторому набору станций соты). Планирование выполняется на основе многих факторов подобно тому, как выполнялось планирование на MAC-уровне в сети LTE. Принимаются во внимание: размер кадра, приоритет QoS и категория трафика, качество подканалов каждой ресурсной единицы, предыстория выделения ресурсных

блоков каждой станции и некоторые другие факторы. В результате вырабатывается план распределения, в котором каждой станции STA выделяется ресурсная единица RU определенного размера (26/52/106/252 тона).

2. Для передачи кадров станциям точка доступа должна выиграть соревнование за доступ к среде с унаследованными станциями, если такие есть в ее соте. Станции стандартов Wi-Fi 6/7 за доступ к среде не соревнуются, а ждут инициативы от точки доступа.
3. После получения доступа точка доступа отправляет кадр физического уровня PPDU, в заголовке которого (заголовок Phy на рис. 25.10) содержится информация о распределении ресурсных элементов между станциями назначения. Эта информация имеет вид пар {AID12, RUn}, где AID12 — это уникальный временный идентификатор станции в пределах данной точки доступа (получен при присоединении), а RUn — порядковый номер ресурсного блока. Заголовок Phy также содержит информацию о структуре ресурсных блоков, то есть о том, каким образом тоны частотного канала объединены в 26, или 52, или 106, или 252 ресурсных блока.

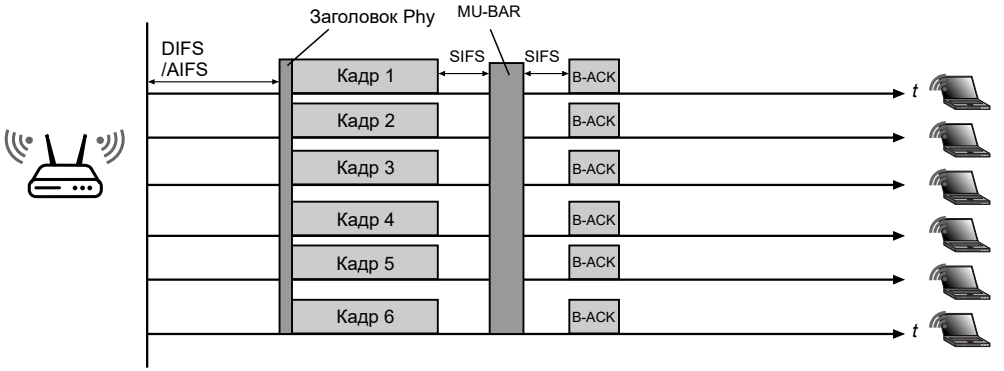


Рис. 25.10. Передача кадров от точки доступа к станциям

Для того чтобы станции смогли прочесть информацию о распределении RU, заголовок Phy передается в радиосреду в технике OFDM. При этом его символы распределены по всему спектру канала 20 МГц (что показано на рисунке сплошной областью, условно покрывшей весь спектр канала). Это естественное решение, так как невозможно послать станциям индивидуальные сообщения в их индивидуальных ресурсных элементах до того, как им сообщили, какой блок принадлежит данной станции.

4. После передачи кадра и паузы SIFS точка доступа посылает всем станциям кадр «Запрос многопользовательского блочного подтверждения» MU-BAR (Multi User Block Ack Request). Этот кадр запрашивает станции, получившие свои кадры при передаче PDDU, подтвердить получение квитанцией «Блоковое подтверждение» Block ACK (блоковое — потому что полученные кадры могли быть агрегированными). Кадр MU-BAR отличается от BAR тем, что он посылается не отдельной станции, а группе станций (используется групповой адрес). При этом для каждой станции в нем указан номер ресурсного блока, в котором станции нужно переслать квитанцию. Согласно логике поведения станций в среде OFDMA, без запроса MU-BAR они не имеют права отправлять свои квитанции. После принятия запроса MU-BAR станции отвечают квитанциями Block ACK в отведенных им ресурсных элементах.

В восходящем направлении UL (рис. 25.11) процесс передачи кадров от станций к точке доступа инициируется отправкой точкой доступа специального управляющего кадра, называемого trigger frame — **триггерный кадр**. В стандартах Wi-Fi 6/7 введено несколько типов триггерных кадров. Все они предназначены для инициирования каких-то действий станций, в данном случае — передачи кадров, и этот тип триггерного кадра называется «Базовый триггер» (Basic Trigger).

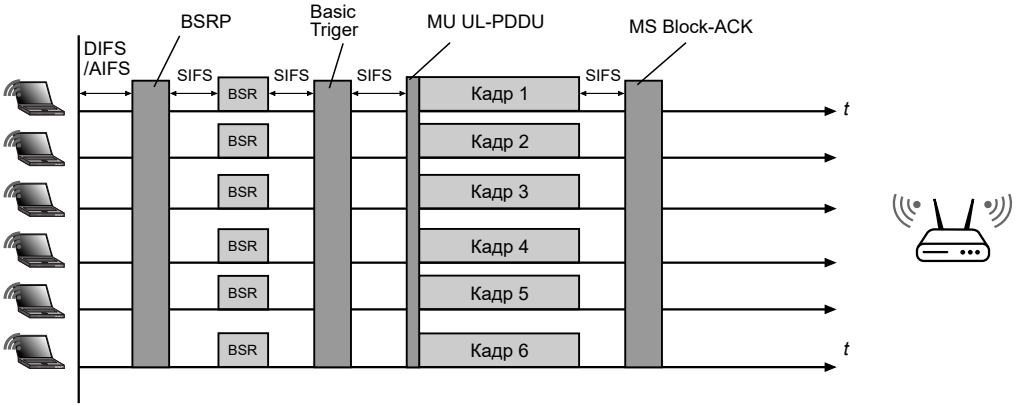


Рис. 25.11. Передача данных несколькими станциями точке доступа

1. Для того чтобы инициировать передачу кадров, точке доступа нужно узнать о потребностях станций в передаче данных. Поэтому точка доступа, получив доступ к среде, предварительно посылает станциям другой триггерный кадр «Запрос состояния буферов» BSRP (Buffer Status Report Poll). Получив его, каждая станция должна отослать индивидуальный ответ в кадре «Отчет о состоянии буфера» BSR (Buffer Status Report). Как и в случае с триггерным кадром BAR, триггерный кадр BSRP передается в режиме OFDM и несет в себе информацию о распределении ресурсных элементов для ответов каждой станции.
2. Каждая станция, найдя себя в списке распределения ресурсных элементов, отвечает кадром BSR, в котором указывает заполненность своего буфера передачи данных, а также категорию QoS, к которой эти данные относятся.
3. Планировщик MAC-уровня точки доступа принимает решение о том, каким станциям нужно предоставить доступ к среде для передачи своих данных, и передает им это решение с помощью триггерного кадра «Базовый триггер» в виде пар {AID12, RUn}. Одновременно передается значение длительности этапа передачи.
4. Станции, которые получили разрешение на передачу своих кадров в определенных ресурсных элементах, делают это после паузы SIFS. Все кадры начинают и заканчивают передаваться в один и тот же момент времени. Те станции, которые закончили передачу раньше установленной длительности передачи, заполняют свой ресурсный элемент битами заполнения (область PAD у кадра 2 на рисунке).
5. Получив кадры от станций, точка доступа отвечает кадром «Блоковое подтверждение для набора станций» (Multi-STA Block ACK), который посылается в режиме OFDM, так что все станции его могут прочитать. В нем точка доступа с помощью битовой карты

подтверждает или не подтверждает получение кадров от станций. На этом цикл передачи данных от станций (направление UL) закончен. Если какой-то станции нужно повторно передать неподтвержденный кадр, то она должна ждать следующего цикла UL.

Циклы DL и UL повторяются, но не в регулярной последовательности. Их очередность зависит от накопленных в буферах данных точки доступа и станций.

Возможно, в будущих версиях Wi-Fi процесс централизованного распределения пропускной способности радиосреды упростится, так как его сложность вызвана необходимостью обратной совместимости со станциями старых стандартов.

Расширенная архитектура сети Wi-Fi

У архитектуры BSS имеется принципиальный недостаток: будучи системой с одной точкой доступа, она плохо масштабируется. Поэтому существует несколько других типов архитектур сетей Wi-Fi, в которых применяется несколько точек доступа.

В первом стандарте 802.11 от 1997 года описана сеть, состоящая из нескольких точек доступа, объединенных проводной сетью. Это сеть названа сетью с расширенным набором услуг — Extended Service Set, ESS. В стандарте предполагается, что распределительная сеть, которая соединяет несколько сот, является сетью канального уровня, например сетью Ethernet, построенной на мостах, или же в современном варианте — на коммутаторах. Так как основной функцией точки доступа является функция моста (а маршрутизатор доступа к интернету — это дополнительный блок точки доступа, встроенный в нее для удобства домашних пользователей), то в целом сеть ESS представляет собой коммутируемую сеть с одним доменом широковещания.

На рис. 25.12 показана ESS, состоящая из двух сетей, BSS 1 и BSS 2, объединенных одним коммутатором Ethernet. Распределительная система может быть более сложной и состоять из нескольких коммутаторов. Она может иметь (и обычно имеет) связь с интернетом через маршрутизатор, подключенный к одному из коммутаторов DS (на рис. 25.12 он не показан).

Каждая из сетей BSS1 и BSS2 имеет одно и то же символьное имя SSID. Станции этих сетей принадлежат одной и той же IP-подсети 192.168.2.0/24. Для исключения интерференции сигналов в области перекрытия зон покрытия сетей им обычно назначают различные частотные диапазоны. Например, если обе точки доступа работают в диапазоне 2,4 ГГц, то сеть BSS1 может работать в канале 1, а сеть BSS2 — в канале 11.

Точки доступа AP1 и AP2 имеют по два порта каждая — беспроводной и проводной. Передача кадра с беспроводного порта на проводной происходит по обычным правилам моста, то есть в том случае, если адрес назначения (Адрес 3 в кадре Wi-Fi, установлен бит «К точке доступа») является широковещательным или же этот адрес точке доступа незнаком (неизученный адрес). При этом точка доступа транслирует кадр из формата Wi-Fi в формат Ethernet, отображая Адрес 3 на адрес назначения, а Адрес 2 — на адрес источника (это адрес STA, пославшей кадр). По классификации мостов точка доступа является *транслирующим мостом* (см. главу 10).

Коммутатор Ethernet, объединяющий сети, работает в стандартном режиме самообучения: он изучает MAC-адреса отправителей всех проходящих через него кадров, поступающих от точки доступа, и строит свою таблицу продвижения.

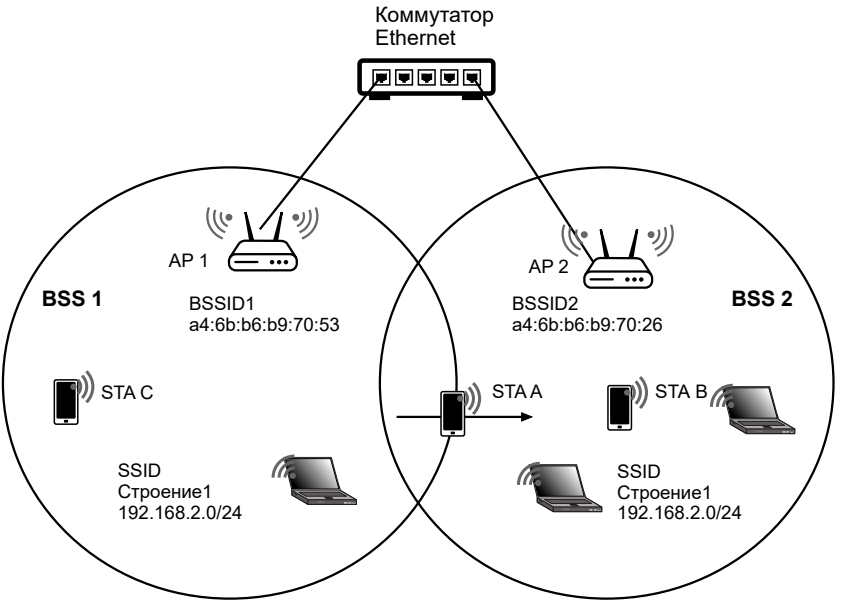


Рис. 25.12. Расширенная сеть ESS

Рассмотрим путь кадра между BSS2 и BSS1 для случая, когда станция STA *B* посылает широковещательный ARP-запрос с IP-адресом станции STA *C*. В этом кадре поля адресов имеют следующие значения:

- Адрес 1 — MAC-адрес точки доступа AP2.
- Адрес 2 — MAC-адрес станции STA *B*.
- Адрес 3 — широковещательный адрес.

Получив этот кадр, точка доступа AP2 передает его в свою радиосреду, а также на порт Ethernet, предварительно преобразовав его в формат Ethernet, — то есть на все свои порты (в соответствии с правилами продвижения кадров с широковещательным адресом). Коммутатор Ethernet передает этот кадр на Ethernet-порт точки доступа AP1, а та, в свою очередь, — в свою радиосреду, предварительно преобразовав его в кадр формата Wi-Fi и поместив в поля адресов следующие значения:

- Адрес 1 — широковещательный адрес назначения Ethernet.
- Адрес 2 — адрес AP1.
- Адрес 3 — адрес STA B (источник) (Адрес 3 является теперь адресом источника, так как установлен бит «От точки доступа»).

Станция STA *C* наряду со всеми станциями сети BSS1 (так как адрес назначения широковещательный) принимает кадр запроса ARP и, видя, что в поле запроса помещен ее IP-адрес, отправляет ARP-ответ точке доступа AP1, поместив его в кадр Wi-Fi, где:

- Адрес 1 — это адрес AP1.
- Адрес 2 — адрес STA C.
- Адрес 3 — адрес STA B.

Точка доступа AP1 изучила, что адрес STA *B* принадлежит проводному порту связи с коммутатором Ethernet, когда передавала широковещательный ARP-запрос от STA *B*, поэтому она передает кадр с ARP-ответом коммутатору Ethernet.

Коммутатор Ethernet также изучил адрес STA *B* при предыдущем продвижении кадра, поэтому он передает его, преобразовав в Ethernet-формат, на порт, к которому подключена точка доступа AP2.

Роуминг в сети ESS

В расширенной сети ESS станция может перемещаться между отдельными сотами BSS, сохраняя связь с сетью ESS, то есть быть *мобильной*. Поддержка мобильности не была первоочередной задачей разработчиков технологии Wi-Fi при ее стандартизации в конце 90-х годов. Они не ставили перед собой задачу сохранения сессий верхнего уровня при переходе из сот в соту, как это сделали разработчики мобильных сотовых телефонных сетей (для последних альтернативы не существовало, задача ставилась именно так). Сеть Wi-Fi позиционировалась как локальная, и большие многосотовые сети Wi-Fi масштаба кампуса или даже города не были целью разработчиков этой технологии.

Тем не менее мобильность в сетях ESS возможна, но она требует полного повторного присоединения станции к новой точке доступа и отсоединения от старой, что занимает большое количество времени, сотни миллисекунд. Эта процедура названа **роумингом** (roaming), а не эстафетной передачей (handover), возможно, чтобы подчеркнуть отличие ее от более совершенной и быстрой процедуры перехода между сотами в мобильных сотовых сетях. Нужно отметить, что со временем были разработаны дополнительные спецификации 802.11k/v/r, которые существенно ускорили роуминг.

Рассмотрим роуминг Wi-Fi (пока без дополнительных спецификаций) на примере сети ESS, показанной на рис. 25.12. Особенностью роуминга Wi-Fi является то, что он осуществляется по инициативе станции, а не точки доступа.

Соты BSS1 и BSS2 частично перекрываются, поэтому при перемещении из соты BSS1 в соту BSS2 станция STA *A* попадает в зону, где она одновременно получает два сигнала маячков точек доступа AP1 и AP2. Мы предполагаем, что соты работают в разных каналах частот, так что сигналы маячков не интерферируют и станция STA *A* их различает. Станция STA *A* также различает соты по MAC-адресам их точек доступа (адреса BSSID), передаваемых в кадрах-маячках.

По мере продвижения из соты BSS1 в направлении соты BSS2 сигнал первой ослабевает, а сигнал второй усиливается. При достижении определенного порога, станция STA *A* решает, что ей нужно переключиться на соту BSS2, после чего она посылает точке доступа AP2 запрос на переприсоединение с указанием адреса ее старой точки доступа AP1. Если точка доступа AP2 отвечает согласием, то станция выполняет полную процедуру присоединения к BSS2, включая повторную аутентификацию, взаимодействуя с точкой доступа AP2. Так как сеть ESS — это одна IP-подсеть, то новый IP-адрес станции STA *A* не нужен. Коммутатор, соединяющий точки доступа AP1 и AP2, передавая кадры между сотами, узнает, что станция STA *A* переместилась из соты в соту и корректирует свою таблицу продвижения.

Для ускорения роуминга были разработаны три спецификации, которые совместно образуют основу для «быстрого роуминга». В соответствии с этими спецификациями, точки

доступа сети ESS «помогают» станции STA выполнить роуминг (хотя инициатива и принятие решений остаются за станцией):

- 802.11k — точка доступа по запросу сообщает станции список соседних точек доступа с характеристиками их частотных каналов. Это ускоряет поиск станцией подходящей соты для роуминга.
- 802.11v — точка доступа сообщает станции дополнительную информацию о загруженности каждой точки доступа из списка 802.11k, тем самым помогая станции выбрать наилучшего кандидата из списка.
- 802.11r — распространяет полученный во время первой аутентификации станции с определенной точкой доступа контекст безопасности между остальными точками доступа. Это позволяет не проводить полную повторную процедуру аутентификации при присоединении станции во время роуминга к новой точке доступа, а заменить ее упрощенной и более быстрой версией.

Примененные вместе, спецификации 802.11k/v/r позволяют сократить время роуминга с сотен миллисекунд до примерно пятидесяти. Но, к сожалению, механизм 802.11r работает (или приносит существенный эффект) не со всеми типами аутентификации, применяемыми в сетях Wi-Fi. Поэтому некоторые производители оборудования Wi-Fi используют фирменные протоколы для быстрого роуминга станций.

Вместо коммутатора, объединяющего сети BSS в сеть ESS, можно применить **маршрутизатор**. Этот вариант используется в крупных сетях Wi-Fi, в которых по определенным причинам необходимо разбивать IP-сеть на подсети (например, из-за того, что в большом домене, построенном только на коммутаторах, возможны широковещательные штормы). В этом случае маршрутизатор соединяет группы точек доступа, принадлежащих разным IP-подсетям. Строго говоря, маршрутизируемая сеть, состоящая из нескольких IP-подсетей, не является сетью ESS. Это сеть IP, объединяющая сети BSS.

При роуминге станции из одной IP-подсети в другую необходимо пройти полную процедуру присоединения к новой точке доступа, с полной аутентификацией, и в процессе этого получить новый IP-адрес из пула адресов новой подсети.

Ячеистая архитектура сети Wi-Fi

Мы рассмотрели архитектуру сети ESS, в которой распределительная сеть DS является проводной сетью канального уровня. Существует возможность построить сеть ESS без проводных соединений, используя только беспроводные связи, и такую возможность дает спецификация 802.11s. Эта спецификация позволяет строить ячеистые сети различной архитектуры, как с инфраструктурой, состоящей из нескольких точек доступа, так и одноранговые бесструктурные сети AdHoc Wi-Fi.

Ячеистая сеть Wi-Fi состоит из узлов нескольких типов:

- **Mesh-узел** (Mesh STA) — станция, поддерживающая стандартную функциональность станции Wi-Fi и дополнительно поддерживающая протоколы создания ячеистой сети и маршрутизации кадров в ней.
- **Mesh-точка доступа** (Mesh-AP) — mesh-узел, который совмещает свои функции с функциями стандартной точки доступа.

- **Mesh-портал** (Mesh Portal) — mesh-узел, который имеет связь с внешней сетью; обычно этот узел связан с корпоративной IP-сетью или же сетью интернет-провайдера.
- Обычная станция **(STA)** стандарта 802.11.

Образование сети происходит распределенно, каждый mesh-узел периодически широковещательно рассылает кадры-маячки таким же образом, каким это делает точка доступа Wi-Fi. В этих кадрах указывается идентификатор mesh-сети, к которой mesh-узел хочет присоединиться, а также характеристики узла — частоты, на которых он работает, и протоколы, которые он поддерживает. Все mesh-узлы, которые находятся в зоне покрытия друг друга, приняв маячковые кадры от соседей по сети, могут установить mesh-связи между собой при условии, что их характеристики совпадают. Установлением связи занимается специальный протокол — Peer Link Management Protocol, PLMP. В процессе обмена сообщениями этого протокола между парой mesh-узлов устанавливается физическая и логическая связь «точка-точка». Одним их этапов установления связи является взаимная строгая аутентификация на основе разделяемого секрета.

На рис. 25.13 показаны семь mesh-узлов, установивших связи между собой и образовавших сеть с Mesh ID «Главный корпус». Не все пары узлов связаны связью «точка-точка». Причины этого могут быть разные: слишком большие расстояния, физические препятствия, помехи. В результате образовалась сеть с ячеистой топологией.

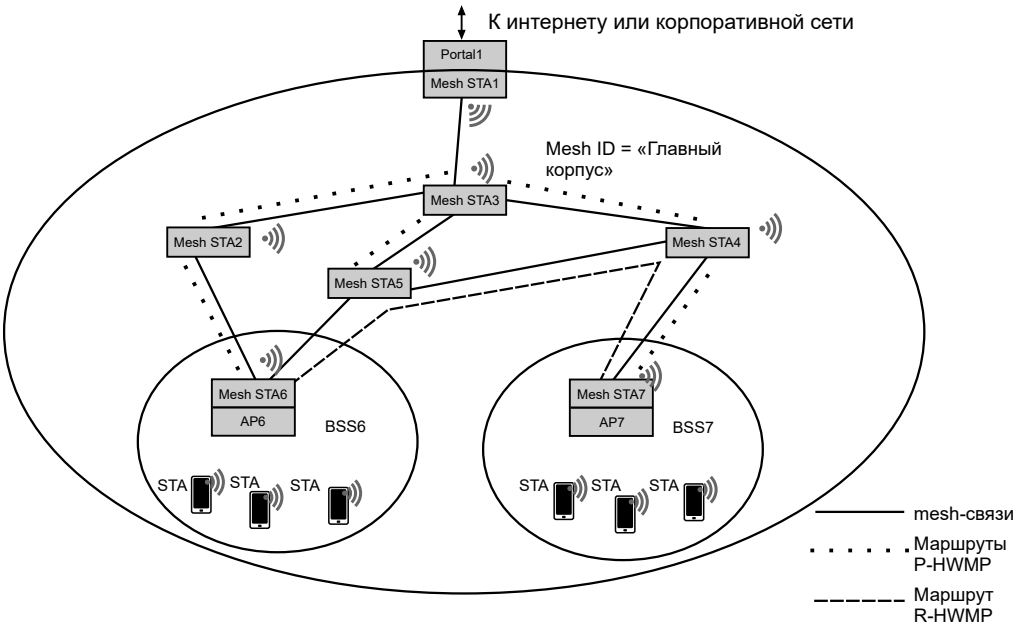


Рис. 25.13. Сеть 802.11s

В этой сети имеются: два mesh-узла, являющихся точками доступа сетей BSS6 и BSS7 (узлы Mesh STA6/AP6 и Mesh STA7/AP7), некоторое количество стандартных станций STA, входящих в сети BSS6 и BSS7, четыре mesh-узла Mesh STA 2–5 и портал Mesh STA1/Portal1. Связи между узлами показаны на рисунке пунктиром. Особенностью протокола

PLMP является то, что даже в случае разрыва связи узел оставляет ее в своей таблице связи в расчете на то, что беспроводная связь может прерываться временно, а потом восстанавливаться.

Маршрутизация в сети 802.11s нужна, потому что это не одношаговая сеть, как BSS, и не сеть с топологией «звезда», как сеть ESS, а сеть с ячеистой топологией. Маршрутизация в сети 802.11s осуществляется на канальном уровне *с использованием MAC-адресов* с помощью протокола Hybrid Wireless Mesh Protocol (**HWMP**) — гибридного беспроводного mesh-протокола.

Определение «гибридный» появилось в названии протокола потому, что он состоит из двух протоколов — *проактивного* и *реактивного*, которые применяются независимо друг от друга для решения различных задач. Проактивный протокол маршрутизации, P-HWMP, применяется для нахождения кратчайших путей от mesh-узла до портала. Реактивный протокол R-HWMP служит для нахождения кратчайших путей между произвольными mesh-узлами сети. Два протокола маршрутизации служат двум разным целям: протокол P-HWMP предназначен для работы в устойчивой конфигурации инфраструктурной ячеистой сети, когда в сети есть портал и основной трафик посылается через портал в интернет или корпоративную сеть. Протокол R-HWMP находит кратчайший путь между любыми двумя mesh-узлами сети по запросу, когда одному mesh-узлу нужно послать кадр другому mesh-узлу, и адреса назначения нет в таблице маршрутизации узла-источника.

С помощью протокола P-HWMP в сети строится древовидная структура маршрутов к корню дерева — порталу. Для этого портал периодически рассылает маршрутные объявления своим соседям, а те распространяют их дальше по сети, наращивая при этом метрику расстояния от себя до корня дерева. Каждый узел выбирает в качестве следующего хопа узел, от которого он получил маршрутное объявление корня с наименьшей метрикой, и помещает его в таблицу маршрутизации. Протокол P-HWMP использует механизм последовательных номеров своих маршрутных объявлений для предотвращения маршрутных петель. Этот протокол в некоторых отношениях похож на протокол Spanning Tree: они оба строят дерево маршрутов на основе объявлений узла — корня дерева. Но есть и отличия: если протокол Spanning Tree работает в сети узлов-коммутаторов, то здесь каждый узел — это маршрутизатор, и порты узлов не блокируются для создания древовидной топологии, а остаются активными и могут передавать кадры, предназначенные для узлов сети, не являющихся порталом.

На рис. 25.13 маршруты, созданные протоколом P-HWMP, показаны пунктирными линиями. Маршруты, созданные протоколом P-HWMP, соединяют все mesh-узлы сети и являются оптимальными для трафика между порталом и mesh-узлами, но они не всегда оптимальны для обмена данными между самими mesh-узлами. Например, mesh-узел STA6 должен обмениваться данными с mesh-узлом STA7 через обходной маршрут P-HWMP через mesh-узел Portal1. Эту проблему решает реактивный протокол маршрутизации R-HWMP. В этом протоколе все mesh-узлы рассматриваются как равные. В исходном состоянии их таблицы маршрутизации пусты, а в том случае, когда некоторому узлу нужно послать кадр другому узлу, он посылает широковещательный запрос своим соседям — не знают ли они пути к узлу назначения.

Пусть в сети нашего примера mesh-узел STA6 хочет послать кадр mesh-узлу STA7. Он посылает широковещательный запрос mesh-узлам STA2 и STA5 с указанием адреса MAC(STA7). Если узлы STA2 и STA5 знают путь к узлу STA7, то они отвечают направленным сообщением об этом узлу STA6, который выбирает лучший маршрут и устанавливают его в своей таблице

маршрутизации, после чего отправляет кадр следующему хопу — STA2 или STA5. Если же они путь к STA7 не знают, то они ретранслируют широковещательный запрос своим соседям — узлам STA3 и STA4, а также узлу STA6, так как он тоже находится в их зоне покрытия и с ним у них имеется mesh-связь. Для предотвращения затопления сети каждый запрос нумеруется узлом — отправителем запроса, так что промежуточные узлы отфильтровывают повторные запросы с одинаковым номером. Наконец, запрос достигает mesh-узла STA7 через mesh-узел STA4. Узел STA7 видит в запросе свой адрес, поэтому он посылает направленный ответ узлу STA4, который в свою очередь направленно пересылает его узлу, от которого он получил запрос первым; будем считать, что это узел STA5 (им мог бы быть и узел STA3). Далее ответ путешествует в обратном направлении, STA5 → STA6, при этом каждый узел наращивает метрику пути, как это обычно делается в дистанционно-векторных протоколах. Узел STA6 устанавливает маршрут к STA7 со следующим хопом STA5 и отправляет кадр для STA7 узлу STA5. Маршруты протокола R-HWMP имеют определенный срок жизни и, если маршрут не подтверждается по истечении этого срока, то он из таблицы маршрутизации удаляется. После чего процесс поиска маршрута к узлу STA7 повторяется.

Периодического затопления сети маршрутными объявлениями от каждого узла, как в проактивном протоколе P-HWMP (или RIP), здесь нет.

Реактивный способ маршрутизации не очень эффективен для ситуаций, когда топология сети стабильна, так как он ведет к задержкам передачи кадров на время поиска маршрута. Но для сети с часто меняющейся топологией он более эффективен, чем проактивный протокол, так как сеть не затапливается маршрутными объявлениями всякий раз, когда происходит изменение топологии, а только в тех случаях, когда нужно передать кадр между узлами сети. В главе 27 вы познакомитесь еще с одним реактивным протоколом маршрутизации — AODV, основанным на аналогичных принципах нахождения маршрута.

Доступ к среде в сети 802.11s происходит в полном соответствии с методами доступа, определенными основными спецификациями 802.11. Все mesh-узлы и стандартные станции STA соревнуются за доступ к среде, используя процедуру CSMA/CA. Выиграв соревнование и получив доступ на период TXOP, станция использует это время для своих нужд, и эти нужды зависят от типа станции.

Стандартная станция STA, присоединенная к некоторой точке доступа, использует период TXOP для передачи своего кадра точке доступа. Стандартная станция не участвует в mesh-активности, поэтому ее поведение не меняется от того, что ее точка доступа является mesh-узлом.

Mesh-узел/точка доступа использует период TXOP либо для передачи кадра присоединенным станциям STA, либо для передачи кадра другому mesh-узлу. В последнем случае интерес представляет передача кадра между mesh-узлами, работающими в стандартах 802.11ax или 802.11be, использующих мультиплексирование OFDMA. При таком мультиплексировании точка доступа определяет распределение ресурсных элементов между станциями, а станции выполняют решение точки доступа, но в данном случае передача идет между двумя равноправными узлами. Решение для нашего случая состоит в том, что узел, выигравший соревнование за среду, играет роль точки доступа и принимает решение о распределении ресурсных элементов, поле чего сообщает его второму mesh-узлу (узлу назначения) в заголовке физического уровня отправленного кадра.

Спецификация 802.11s позволяет строить **одноранговые сети AdHoc Wi-Fi** в том случае, когда все узлы сети сконфигурированы как mesh-узлы с общим разделяемым паролем

и общим идентификатором Mesh ID. Стандартных станций STA и портала в такой сети нет, при этом применяется только один протокол маршрутизации R-HWMP.

Такая одноранговая AdHoc-сеть соответствует сети типа IBSS (независимой BSS), описанной в ранних стандартах сетей Wi-Fi. Но там такая архитектура была описана в очень общих чертах, только как идея. Спецификация 802.11s позволяет эту идею воплотить в качестве законченной технологии.

Необходимо отличать точки доступа, поддерживающие mesh-технологию (иногда в ее нестандартном варианте, обеспечивающем те же свойства, но не всегда на основе протоколов спецификации 802.11s) от **расширителей сети Wi-Fi**, называемых также повторителями, или **Wi-Fi Extenders**. Расширитель Wi-Fi — это точка доступа с усеченными функциями. В качестве точки доступа она присоединяет к своей сети станции по стандартной процедуре присоединения. Но она является подчиненным узлом основной точки доступа сети, к которой сама должна присоединиться как станция. Ее подчиненность проявляется в том, что любой кадр, получаемый ею от своей станции, она передает основной точке доступа, а не станции назначения. После этого основная точка доступа возвращает этот кадр расширителю, а в свою очередь расширитель передает кадр станции назначения. Если расширитель использует тот же частотный диапазон, что и основная точка доступа, то один и тот же кадр передается через разделяемую среду не два раза (вариант без расширителя), а четыре — что приводит к уменьшению пропускной способности среды в 2 раза. Основное достоинство расширителя — увеличение области покрытия сети. Расширитель может работать в двух частотных диапазонах, используя, например, диапазон 2,4 ГГц для связи со станциями, а диапазон 5 ГГц — исключительно для связи с основной точкой доступа. В этом случае уменьшения пропускной способности среды не происходит.

Технология Bluetooth

Особенности персональных сетей

Персональные сети (Personal Area Network, **PAN**) предназначены для взаимодействия устройств, принадлежащих одному владельцу, на небольшом расстоянии, обычно в радиусе 10 метров. Технология Bluetooth была первой стандартизованной технологией этого типа.

Типичным примером PAN является беспроводное соединение компьютера с периферийными устройствами: принтер, наушники, мышь, клавиатура и т. п. Мобильные телефоны также используют технологию PAN для соединения со своей периферией (чаще всего это наушники), а также с компьютером своего владельца. Персональные сети во многом похожи на локальные, но у них есть и свои особенности:

- *Область покрытия PAN меньше области покрытия LAN.* Узлы PAN часто находятся на расстоянии нескольких метров друг от друга.
- *Высокие требования к безопасности.* Персональные устройства, путешествуя вместе со своим владельцем, попадают в различное окружение. Иногда они должны взаимодействовать с устройствами других персональных сетей, например, если их владелец встретил на улице своего знакомого и решил переписать из его телефона в свой несколько

адресов общих знакомых. В других случаях такое взаимодействие явно нежелательно, так как может привести к утечке конфиденциальной информации. Поэтому протоколы PAN должны обеспечивать разнообразные методы аутентификации устройств и шифрования данных в мобильной обстановке.

- При соединении в сеть бытовых приборов или малогабаритных устройств желание избавиться от кабелей проявляется гораздо сильнее, чем при соединении компьютера с принтером или концентратором. Из-за этого персональные сети в гораздо большей степени, чем локальные, *тяготеют к беспроводным решениям*.
- Если человек постоянно носит устройство PAN с собой или на себе, то оно не должно причинять вред его здоровью. Поэтому такое устройство должно *излучать сигналы небольшой мощности*, желательно не более 100 мВт (обычный сотовый телефон излучает сигналы мощностью 200–1000 мВт).
- И наконец, исходя из областей применения, можно сформулировать еще одно требование — эта технология не должна быть дорогой.

Сегодня самой популярной технологией PAN является Bluetooth, которая обеспечивает взаимодействие восьми устройств в разделяемой среде диапазона 2,4 МГц с битовой скоростью передачи данных до 50 Мбит/с.

Архитектура Bluetooth

Стандарт **Bluetooth** разработан группой **IEEE 802.15.1** в соответствии с общей структурой стандартов IEEE 802. В технологии Bluetooth используется термин **пикосеть** (piconet, pico — маленький), который подчеркивает небольшую область покрытия — от 10 до 100 метров (в зависимости от мощности излучения передатчика устройства). Пикосеть Bluetooth относится к категории сетей AdHoc — она создается произвольным образом, на относительно короткое время. В пикосеть может входить до 255 устройств, но только восемь из них могут в каждый момент времени быть *активными* и обмениваться данными. Одно из устройств в пикосети является *главным*, остальные — *подчиненными* (рис. 25.14). (На рисунке также показана распределенная пикосеть, имеющая несколько главных узлов; о ней будет сказано позже.)

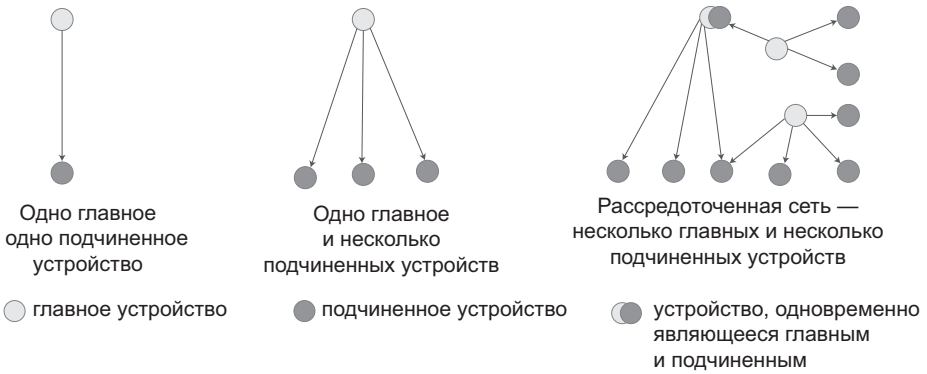


Рис. 25.14. Пикосеть и рассредоточенная сеть

Активное подчиненное устройство может обмениваться данными только с главным устройством (прямой обмен между подчиненными устройствами невозможен). Все устройства пикосети, не являющиеся активными, должны находиться в режиме пониженного энергопотребления, в котором они только периодически прослушивают команду главного устройства для перехода в активное состояние.

Главное устройство отвечает за доступ к *разделяемой среде пикосети*, представляющей собой нелицензируемые частоты диапазона 2,4 ГГц. Пропускная способность среды делится главным устройством между семью подчиненными устройствами на основе техники разделения времени TDM. Такая архитектура позволяет применять более простые протоколы в устройствах, выполняющих функции подчиненных (например, в радионаушниках), и отдает более сложные функции управления пикосетью компьютеру, который обычно является главным устройством сети.

Присоединение к пикосети происходит динамически. Главное устройство пикосети, используя процедуру опроса, собирает информацию об устройствах, попадающих в зону его покрытия. Обнаружив новое устройство, главное устройство проводит с ним переговоры. Если желание подчиненного устройства присоединиться к пикосети совпадает с решением главного устройства (подчиненное устройство прошло проверку аутентичности и оказалось в списке разрешенных устройств), то новое подчиненное устройство присоединяется к сети. Безопасность сетей Bluetooth обеспечивается за счет аутентификации устройств и шифрования передаваемого трафика.

Производитель оборудования Bluetooth может определить роль конкретного устройства — главное или подчиненное — за счет задания ее в программной реализации стека его протоколов. Однако эта роль может измениться в ходе переговоров при установлении соединения и главное устройство может стать подчиненным, если выяснится, что подчиненное устройство обладает более развитой функциональностью и способно поддерживать обе роли.

Несколько пикосетей, которые обмениваются между собой данными, образуют **рассредоточенную** сеть. Взаимодействие в пределах рассредоточенной сети осуществляется за счет того, что один узел (называемый мостом) одновременно является членом нескольких пикосетей, причем этот узел может исполнять роль главного устройства одной пикосети и подчиненного устройства другой.

В сетях Bluetooth для передачи информации используются разные методы:

- *Для чувствительного к задержкам трафика* (например, голоса) сеть поддерживает **синхронный канал, ориентированный на соединение** (Synchronous Connection-Oriented link, **SCO**). Этот канал работает на скорости 64 Кбит/с. Для канала SCO пропускная способность резервируется на все время соединения.
- *Для эластичного трафика* (например, компьютерных данных) используется работающий с переменной скоростью **асинхронный канал, не ориентированный на соединение** (Asynchronous Connection-Less link, **ACL**). Для канала ACL пропускная способность выделяется по запросу подчиненного устройства или по потребности главного устройства.

Bluetooth является законченной оригинальной технологией, рассчитанной на самостоятельное применение в электронных персональных устройствах и поддерживающей **полный стек протоколов**, включая собственные прикладные протоколы. В этом заключается ее отличие от рассмотренных ранее технологий, таких как Ethernet или Wi-Fi, которые лишь выполняют функции физического и канального уровней.

Создание для технологии Bluetooth собственных прикладных протоколов объясняется стремлением разработчиков реализовывать ее в разнообразных простых устройствах, которым не под силу, да и не к чему, поддерживать стек протоколов TCP/IP. Технология Bluetooth появилась в результате попыток разработать стандарт для взаимодействия мобильного телефона с беспроводными наушниками. Понятно, что для решения такой простой задачи не нужен ни протокол передачи файлов (FTP), ни протокол передачи гипертекста (HTTP). В результате для технологии Bluetooth был создан оригинальный стек протоколов, в дополнение к которому появилось большое количество профилей. **Профили** определяют конкретный набор протоколов для решения той или иной задачи. Например, существует профиль для взаимодействия компьютера или мобильного телефона с беспроводными наушниками. Имеется также профиль для тех устройств, которые могут передавать файлы (наушникам он, скорее всего, не потребуется), профиль эмуляции последовательного порта RS232 и т. д.

Поиск и стыковка устройств Bluetooth

Устройство, поддерживающее технологию Bluetooth, обычно посылает периодические запросы на предмет обнаружения других устройств Bluetooth в зоне досягаемости. Устройство Bluetooth, получившее запрос и сконфигурированное так, чтобы отвечать на запросы, в ответ передает сведения о себе: имя и тип устройства, имя производителя, поддерживаемые сервисы. Имя устройства конфигурируется, в отличие от его уникального MAC-адреса, который задается производителем. Отметим, что часто устройства выпускаются со сконфигурированными по умолчанию именами, соответствующими названию модели устройства, поэтому в сфере досягаемости вашего мобильного телефона может оказаться несколько других телефонов с одинаковыми именами Bluetooth, если их владельцы не дали им собственные имена.

После предварительного обмена информацией устройства Bluetooth могут начать так называемую процедуру стыковки (pairing), если конфигурация устройств ее требует. Стыковка подразумевает установление защищенного канала (см. главу 32) между устройствами; безопасность в данном случае означает, что устройства доверяют друг другу, а данные между ними передаются в зашифрованном виде. Стыковка устройств Bluetooth требует введения в каждое из них одного и того же пароля, называемого также PIN-кодом Bluetooth. Обычно устройство, получившее запрос на стыковку, просит пользователя ввести PIN-код. Устройства, успешно прошедшие процедуру стыковки, запоминают этот факт и устанавливают безопасное соединение автоматически всякий раз, когда оказываются в зоне досягаемости, при этом повторное введение PIN-кода пользователем не требуется. Устройство сможет быть сконфигурировано пользователем (производителем) таким образом, чтобы разрешать установление соединений с другими устройствами без процедуры стыковки.

Физический уровень Bluetooth

Сеть Bluetooth использует технику расширения спектра скачкообразной перестройкой частоты FHSS в диапазоне частот 2,4 ГГц. Чтобы сигналы разных пикосетей не интерферировали, каждое главное устройство использует собственную последовательность псевдослучайной перестройки частоты. Наличие различающихся последовательностей

псевдослучайной перестройки частоты затрудняет общение узлов, принадлежащих разным пикосетям в рассредоточенной сети. Для преодоления этой проблемы устройство-мост должно при подключении к каждой из пикосетей соответствующим образом менять последовательность.

Каждый частотный канал FHSS имеет ширину 1 МГц, количество каналов равно 79 (в России, США и большинстве других стран) или 23 (в Испании, Франции, Японии). Скорость изменения частоты канала равна 1600 Гц. Главное устройство пикосети делит между подчиненными станциями общую среду на основе временного мультиплексирования TDM, используя в качестве тайм-слота время пребывания системы на одном частотном канале, то есть 1/1600 = 625 мкс. В течение одного тайм-слота пикосеть Bluetooth передает данные со скоростью 1 Мбит/с, что дает 625 бит за один тайм-слот, но не все биты служат для передачи полезной информации — при смене частоты устройствам сети требуется некоторое время для синхронизации, поэтому из 625 бит только 366 передают данные кадра.

Стандарт Bluetooth не определяет конкретной процедуры распределения тайм-слотов главным устройством пикосети между подчиненными устройствами, оставляя ее детали на усмотрение производителей оборудования. На практике реализуются несколько **алгоритмов планирования распределения тайм-слотов**.

- **Опрос**. Главный узел периодически опрашивает подчиненные узлы для выяснения их потребностей в передаче и выделяет очередной тайм-слот одному из них, сообщая также и номер частотного канала, на котором нужно вести передачу.
- **Круговое распределение**. Каждый подчиненный узел получает право передавать свои данные по фиксированной очереди. Это простое, но неэффективное решение, если интенсивность трафика от подчиненных узлов существенно неравномерная.
- **Динамическое распределение тайм-слотов с учетом требований QoS.** Наиболее гибкий способ, когда главный узел выясняет потребности подчиненных узлов с учетом требований QoS и затем выделяет тайм-слоты (возможно, несколько подряд) одному из них. Наиболее эффективный алгоритм, но в то же время и наиболее сложный в реализации.

Устройства Bluetooth делятся на три класса в зависимости от мощности передатчика и предполагаемой дальности его работы (табл. 25.2).

Таблица 25.2. Классы устройств Bluetooth

Класс[1]	Мощность передатчика	Дальность[2]
Класс 3	1 мВт	< 10 м
Класс 2	2,5 мВт	10 м
Класс 1	100 мВт	100 м

[1] Большинство бытовых устройств Bluetooth относятся ко 2 классу.

[2] Дальность указана предполагаемой, поскольку не учитывается наличие препятствий между передатчиком и приемником и уровня помех в радиосреде. На практике она может оказаться меньше заявленной.

Физический уровень стека протоколов Bluetooth постоянно совершенствуется. Версия 1.0 стандартов стека была принята в 1999 году, а последняя на момент написания данной книги версия 6.2 принята в конце 2025 года.

В версии Bluetooth 1.0 информация кодируется с тактовой частотой 1 МГц путем двоичной частотной манипуляции (BFSK) — в результате битовая скорость составляет 1 Мбит/с.

В версии 2.0 введен режим **улучшенной скорости передачи данных** (Enhanced Data Rate, EDR), в котором для кодирования данных используется комбинация методов частотной (BFSK) и фазовой (PSK) модуляции, позволивших повысить битовую скорость до 3 Мбит/с, а полезную скорость передачи данных — до 2,1 Мбит/с. Режим EDR дополняет основной режим передачи данных со скоростью 1 Мбит/с. Кадр данных может занимать 1, 3 или 5 слотов. Когда кадр занимает больше одного слота, частота канала остается неизменной в течение всего времени передачи кадра. В этом случае накладные расходы на синхронизацию меньше, так что размер кадра, состоящего, например, из пяти последовательных слотов, равен 2870 бит (с полем данных — до 2744 бит).

В апреле 2009 года группа Bluetooth SIG объявила о новом дополнительном стеке протоколов под названием **Bluetooth Low Energy (Bluetooth LE)**. Протоколы Bluetooth LE предназначены для устройств, батареи которых должны иметь примерно годичный срок действия; это могут быть, например, наручные часы или медицинские приборы. Технология Bluetooth LE получила маркетинговое название **Bluetooth Smart** и сегодня реализована в большинстве смартфонов и планшетов. Существуют реализации протоколов Bluetooth Smart в качестве единственного стека протоколов Bluetooth некоторого устройства, а также в варианте второго стека протоколов, работающего наряду с классическим стеком, — в этом случае такое устройство маркируется как Bluetooth Smart Ready. Для передачи данных Bluetooth LE использует тот же частотный диапазон 2,4 ГГц, что и классический вариант Bluetooth, но в нем организуется не 79 каналов с полосой 1 МГц, а 40 каналов с полосой 2 МГц каждый. Битовая скорость передачи данных Bluetooth LE составляет 1 Мбит/с. Передача голоса по Bluetooth LE не предусмотрена.

Спецификация Bluetooth LE описана в версии Bluetooth SIG 4.0 стандарта, а в версии Bluetooth 5 вводятся некоторые ее усовершенствования:

- Скорость передачи данных увеличена до 2 Мбит/с (при этом скорость передачи пользовательских данных возросла в 1,7 раза, так как интервалы между кадрами данных остались прежние).
- Введен специальный **режим Long Distance** (LD) — большие расстояния, — предназначенный для объектов интернета вещей, которым нужно взаимодействовать на расстояниях, больших чем 100 м, и при этом соблюдать режим энергосбережения своей батареи. Увеличение расстояния достигается за счет уменьшения битовой скорости передачи данных в два раза, по сравнению с Bluetooth LE, то есть до 1 Мбит/с, и применения самокорректирующихся кодов FEC. Существует две опции применения кодов FEC. В первой опции каждый исходный бит данных заменяется двумя битами результирующего кода FEC. При этом скорость пользовательских данных уменьшается в два раза по сравнению со скоростью передачи данных, то есть равна 500 Мбит/с, а максимальное расстояние увеличивается примерно вдвое, то есть до 200 м. Во второй опции каждый исходный бит данных заменяется 8 битами кода FEC, скорость пользовательских данных уменьшается в 8 раз, то есть становится равной 125 Мбит/с, но зато максимальное расстояние увеличивается в 4 раза, то есть примерно до 400 метров.

Сосуществование Bluetooth и Wi-Fi

Bluetooth использует тот же диапазон частот, что и Wi-Fi, — диапазон 2,4 ГГц. Мощность передатчика узла Wi-Fi намного выше, чем узла Bluetooth, — 100 мВ против 2,5 мВ, поэтому в том случае, если они работают на одной и той же частоте, то сигнал Bluetooth просто подавляется более мощным сигналом Wi-Fi. Чтобы этого избежать, в технологии Bluetooth немного изменен алгоритм работы метода скачкообразной перестройки частоты FHSS. Узел Bluetooth всегда проверяет радиосреду, чтобы определить, не работает ли в каком-либо канале диапазона 2,4 ГГц передатчик Wi-Fi. И если канал занят, то узел Bluetooth исключает свои частоты, попадающие в канал Wi-Fi, и использует в скачкообразных переходах только оставшиеся. Так как ширина канала Wi-Fi равна 22 мГц, а канала Bluetooth — 1 МГц, то исключается довольно много каналов: из 79 остается 55 (охранные диапазоны между каналами уменьшают их количество). Такой способ сосуществования называется *адаптивной скачкообразной перестройкой*.

ГЛАВА 26 Беспроводные технологии с низким энергопотреблением

Мы уже обсудили в разделе «Ограничения энергопотребления» главы 22, какую важную роль играет низкое энергопотребление в беспроводных сетях. В этой главе мы изучим несколько технологий, специально разработанных для обеспечения этого свойства. Такие технологии, называемые **беспроводными технологиями с низким энергопотреблением НЭ** (Low Power, LP), обеспечивают срок службы батареи от 5 до 15 лет, поддерживая при этом скорость передачи данных от нескольких бит в секунду до нескольких сотен килобит в секунду. Работать такие технологии могут на различных расстояниях, от нескольких метров (НЭ-технологии близкого радиуса действия) до нескольких километров (НЭ-технологии дальнего радиуса действия).

Технологии НЭ в последнее время стали одной из наиболее быстро развивающихся областей сетевых технологий. Основной причиной такой ситуации является растущая популярность интернета вещей с его миллионами простых узлов с очень ограниченными энергетическими и вычислительными ресурсами. Но потребность в технологиях НЭ не ограничивается сетями интернета вещей. Имеются и другие области деятельности человека, где требование энергосбережения является жизненно необходимым, например сети дронов, рассматриваемые в главе 27.

Растущую актуальность технологий НЭ подтверждают цифры их внедрения: к концу 2023 года в мире насчитывалось около 1,3 миллиарда узлов, подключенных к интернету с помощью технологий НЭ. По прогнозу ежегодный рост числа таких узлов будет составлять в среднем 27 %, так что к концу 2027 года их будет насчитываться около 3 миллиардов.

Области применения

Прежде всего отметим, что беспроводные НЭ-технологии могут эффективно работать только в сетях, в которых передаются *небольшие объемы данных*. Это связано с тем, что беспроводной узел расходует свою электроэнергию *наиболее интенсивно во время радиопередачи*. Хотя вычисления, производимые контроллером узла, также требуют потребления энергии батареи, радиопередача потребляет энергии в десятки и сотни раз больше, чем вычисления. Например, для типичного устройства интернета вещей — датчика температуры в доме — при выполнении его микроконтроллером вычислений потребляются микроамперы тока, а при радиопередаче результата — десятки миллиампер. Энергопотребление почти линейно зависит от длительности передачи данных, количества передаваемых пакетов и количества повторов передач пакетов из-за потерь или искажений данных. Например, если узел передает 20 байт телеметрии один раз в день, то его радиопередатчик работает около 50 мс в день, а если же этот узел должен передавать раз в день 20 кбайт данных —

то его радиопередатчик будет работать 5 секунд каждый день. Разница в службе батареи составит несколько лет против нескольких месяцев автономной работы.

Для экономии энергопотребления должна минимизироваться доля времени, в течение которого устройство находится в состоянии передачи. Поэтому передача данных должна происходить не потоковым способом (с заданным периодом повторения), а *при наступлении определенных событий*, например, при превышении температуры окружающей среды заданного порога или возникновении аварийного состояния контролируемого объекта. Каждый раз после передачи небольшой порции данных протоколы технологий НЭ переводят радиопередатчик в спящий режим, в котором он находится до наступления очередного события.

Высокие скорости передачи данных сокращают время передачи и, казалось бы, должны вести к уменьшению энергозатрат узла. Однако это не так: передача данных с высокой скоростью требует повышения мощности радиопередатчика, чтобы обеспечить необходимое значение отношения мощности полезного сигнала к мощности помех SNR — это следует из формулы Шеннона. В результате энергозатраты для передачи одного бита информации на высоких скоростях оказываются выше, чем на низких, и поэтому технологии НЭ выбирают *низкие скорости передачи данных*.

Таким образом, для эффективной работы технологии НЭ в беспроводной сети функционирование узлов этой сети должно соответствовать следующим неформальным условиям: во-первых, данные должны передаваться небольшими порциями, во-вторых — на низкой скорости, в-третьих, передача должна выполняться редко. Если такие условия соблюдаются, узел может работать без замены батареи 10 лет, а сеть может включать десятки тысяч узлов и работать без необходимости вмешательства обслуживающего персонала.

Приведем несколько примеров областей применения технологий НЭ:

- Системы измерения параметров окружающей среды на основе многочисленных датчиков влажности, температуры, давления и т. п., установленных в полевых условиях на обширных пространствах, в лесах или сельскохозяйственных угодьях. Многочисленные датчики такой системы изредка посылают небольшие сообщения в центральный контроллер системы через беспроводные шлюзы, проводя бóльшую часть времени в неактивном состоянии и пробуждаясь только для очередного измерения своего параметра среды и передачи измеренного значения. Скорость передачи таких данных может быть низкой, задержки на качество системы мониторинга не влияют, а вот область покрытия должна быть большой и очень желательно, чтобы датчики могли работать без присмотра длительное время.
- Системы контроля магистральных трубопроводов, линий электрических электропередач и других протяженных промышленных объектов.
- Объекты умного города, требующие контроля и управления, такие как умное уличное освещение, датчики занятости парковочных мест, мониторинг мусорных контейнеров, счетчики трафика и пешеходов. Они менее требовательны к длительности службы батарей, так как их проще периодически обслуживать.
- Объекты коммунального хозяйства: счетчики электроэнергии и расхода газа в домах, счетчики потребления тепла.
- Объекты промышленного интернета вещей: датчики мониторинга и диагностики состояния оборудования (но не датчики и исполнительные механизмы управления обо-

рудованием в реальном времени, потому что эти объекты требуют высокой пропускной способности и низких задержек).

- Объекты умного дома менее требовательны к сроку службы батарей, так как находятся под наблюдением жильцов дома и имеют близкий радиус действия.

Спектр технологий с низким энергопотреблением

Перечень технологий с низким энергопотреблением насчитывает более десятка технологий той или иной степени зрелости. Такая ситуация объясняется разнообразием требований, предъявляемых к таким технологиям. Действительно, требования к связи с командным центром беспилотного летательного аппарата (дрона) очевидно отличаются от требований к связи умного холодильника с телефоном его владельца или датчика влажности, установленного на пшеничном поле и передающего данные один раз в час в центральную систему мониторинга окружающей среды.

Создать технологию, настолько гибкую, чтобы она могла одинаково хорошо работать и на коротких и на длинных расстояниях, с высокой и с низкой скоростями, потреблять мало энергии и обеспечивать низкие задержки, практически невозможно. Перечисленные свойства противоречат друг другу и улучшение одного или нескольких из них часто ведет к ухудшению другого или других.

Например, высокоскоростная технология, обеспечивающая передачу данных на большие расстояния, неизбежно требует применения мощных передатчиков, так как высокой скорости можно достичь только на высокой частоте несущей (следствие эмпирического правила 5 % ширины полосы от частоты несущей, см. главу 23), а высокочастотный сигнал быстро затухает с расстоянием, так что без мощного передатчика не обойтись. А если вы выбрали низкочастотную несущую, обеспечивающую большое расстояние и малое энергопотребление, то никак не получится достичь высоких скоростей передачи. И наконец, низкое энергопотребление и высокие скорости существенно сокращают зону покрытия сети, что мы видим в технологиях персональных сетей.

В этой главе мы рассмотрим четыре технологии класса НЭ: ZigBee, Thread, Wi-SUN FAN и LoRaWAN.

Первые две технологии предназначены для объединения узлов на небольших расстояниях, от 10 до 100 метров, так что они применяются для автоматизации умных домов, офисных зданий и промышленных предприятий. ZigBee является ветераном технологий класса НЭ. Ей насчитывается уже более 20 лет, и 15–30 % (по разным оценкам) устройств умного дома являются сегодня устройствами ZigBee. Thread — это ее молодой конкурент, но быстро догоняющий, имеющий более современный стек протоколов, построенный вокруг IPv6. Обе технологии делят эту область применения с Wi-Fi и Bluetooth по понятной причине — не всем устройствам умного дома нужна энергосберегающая технология, так как многие из них, например телевизор или холодильник, питаются не от батарей, к тому же некоторым из них нужны более высокие скорости передачи данных, чем у технологий НЭ, например, тому же телевизору.

Технологии LoRaWAN и Wi-SUN FAN рассчитаны на создание сетей с большой областью покрытия, в которых расстояние между узлами сети измеряется километрами. LoRaWAN

является сегодня бесспорным лидером в области автоматизации городской инфраструктуры (светофоров, уличных фонарей, датчиков парковки, датчиков расхода электроэнергии), контроля окружающей среды в сельском хозяйстве и некоторых других. Ее инсталляционная база к 2024 году составила 350 миллионов конечных узлов — около четверти всех узлов, использующих технологи НЭ. Технология Wi-SUN FAN занимает гораздо более скромное положение в отношении внедренных систем, поскольку имеет более узкую область применения — автоматизацию протяженных коммунальных сетей, таких как сети электропередач и магистральные трубопроводы, но она интересна тем, что отличается большой гибкостью и надежностью, а также имеет современный стек протоколов на основе протокола IPv6.

Общая платформа 802.15.4

Мы будем рассматривать технологии ZigBee, Thread и Wi-SUN FAN во взаимосвязи, так как эти три технологии построены на общей платформе — **спецификации IEEE 802.15.4**, которая разработана группой IEEE, занимающейся низкоскоростными версиями технологий персональных сетей.

Отличительной особенностью спецификации 802.15.4 является то, что она не определяет законченную технологию, а скорее представляет собой конструктор, позволяющий построить конкретную технологию из имеющихся строительных блоков. Именно такой подход был использован тремя технологиями класса НЭ, рассматриваемыми в этом разделе. За счет определенного выбора режимов и параметров общей платформы разработчикам удалось создать технологии с характеристиками, соответствующими требованиям областей их применения — по скорости передачи данных и максимальному расстоянию между узлами.

Основные составляющие и параметры спецификации IEEE 802.15.4 показаны на рис. 26.1.

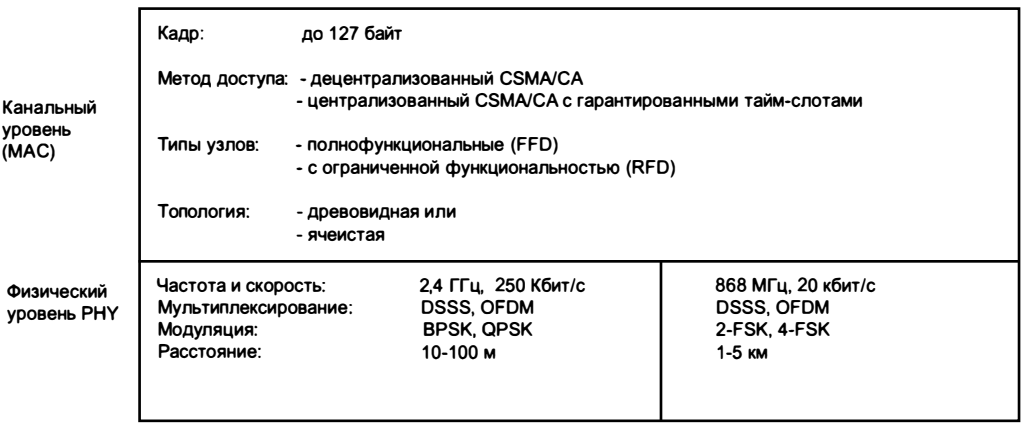


Рис. 26.1. Стек протоколов спецификации IEEE 802.15.4

На *физическом уровне* определены два диапазона частот, на которых может работать узел сети стандарта 802.15.4, — диапазон 2,4 ГГц для более скоростных версий (до 250 Кбит/с) и диапазон 868 МГц для менее скоростных версий (до 20 Кбит/с). Мультиплексирование пользовательских потоков данных в обоих частотных диапазонах может выполняться на основе как техники прямого расширения спектра DSSS, так и техники OFDM. Мощность

передатчика узла 802.15.4 обычно лежит в пределах 0–10 дБм (1–10 мВ), что намного ниже, чем у станций Wi-Fi (100 мВ), так что это не только низкоскоростная технология, но и технология низкого энергопотребления.

На *канальном уровне* платформа 802.15.4 предоставляет следующие функции:

- Адресация узла.
- Обнаружение соседей.
- Присоединение и отсоединение узла к/от сети.
- Анализ качества беспроводных связей.
- Передача кадров между узлами.

Размер кадра канального уровня ограничен 127 байтами. Это сравнительно небольшой размер, который соответствует требованиям технологий НЭ.

Узлы используют на канальном уровне два типа адресов: *постоянный глобальный 64-битный MAC-адрес* (формат EUI-64), который позволяет иметь в сети до 65 537 узлов (гораздо более масштабируемая сеть по сравнению с Bluetooth), и короткий локальный *16-битный временный адрес*, который более эффективен для помещения в 127-байтный кадр и используется после присоединения узла к сети (вы можете вспомнить адрес AID-12 в технологии Wi-Fi, также являющийся временным и локальным в пределах соты).

В сети 802.15.4 определены узлы двух типов:

- **Полнофункциональные** узлы (Full-Function Device, **FFD**), которые могут выполнять в сети разнообразные функции, например, управлять формированием сети — присоединением узлов с их аутентификацией, маршрутизацией данных.
- Узлы **с ограниченной функциональностью** (Reduce-Function Devices, **RFD**), предназначенные для простых устройств, например датчиков, которые имеют ограниченные вычислительные ресурсы и могут поэтому выполнять функции только конечных узлов сети, но не маршрутизаторов или ретрансляторов данных.

Помимо деления узлов сети на полнофункциональные и с ограниченной функциональностью, узлы могут делиться в зависимости от роли, которую они играют в работе сети:

- PAN-координатор.
- Координатор.
- Конечный узел.

PAN-координатор отвечает за формирование сети. Он управляет процессами присоединения и отсоединения узла к/от сети и делает это с помощью периодической отправки в радиосреду кадров-маячков, в которых объявляет идентификатор сети, используемые частотные каналы, а также некоторые другие параметры сети. PAN-координатор назначает узлам сети локальные 16-битные адреса и может взаимодействовать с любым узлом сети. Для выполнения своих функций PAN-координатор всегда должен быть активен, поэтому на практике он обычно подключен к центральной электросети. В сети может быть только единственный узел, играющий роль PAN-координатора, и он должен быть активирован первым. PAN-координатор выполняет функции, близкие к функциям точки доступа сети Wi-Fi.

Координатор сети отвечает за релейную передачу кадров данных между узлами сети. Он может маршрутизировать трафик, если в сети работает протокол маршрутизации. Координатор также может участвовать в процедуре присоединения узлов к сети под управлением PAN-координатора — новый узел может присоединяться не только к PAN-координатору, но

и к координатору, при этом PAN-координатор участвует в этом процессе, назначая локальные адреса присоединяющимся узлам. Координатор, как и PAN-координатор, должен быть всегда активен и поэтому не должен питаться от батареи. В сети может быть несколько узлов-координаторов.

Конечный узел может присоединяться только к PAN-координатору или координатору, и другие узлы к нему присоединяться не могут. Релейные передачи он также выполнять не может: он может только обмениваться данными с узлом, к которому он присоединился. Конечный узел находится в активном состоянии на короткое время, требуемое для обмена очередной порцией данных. Предполагается, что конченый узел питается от батареи.

Соотношение между типами узлов и их ролями показано в табл. 26.1.

Таблица 26.1. Соотношение между типами узлов и их ролями

	PAN-координатор	Координатор	Конечный узел
Полнофункциональный узел	+	+	+
Узел с ограниченной функциональностью	-	-	+

В стандарте 802.15.4 не определяется соответствие между управляемыми объектами сети («вещами»), с одной стороны, и типами и ролями узлов сети, с другой стороны. Например, электрическая лампочка может быть полнофункциональным узлом и играть роль координатора. Датчик задымленности, питающийся от батареи, как правило, является конечным узлом. PAN-координатором обычно является специализированное устройство. Подробнее мы поговорим об этом, рассматривая пример сети ZigBee в конце этого раздела.

Спецификация 802.15.4 поддерживает топологии двух типов — древовидную и ячеистую.

Древовидная топология имеет один корень и произвольное число уровней — ветвей дерева, к которым присоединены листья. Корнем дерева является PAN-координатор, координаторы образуют ветви дерева, а листьями являются конечные узлы. В соответствии со своими функциями PAN-координатор начинает образование сети, объявляя с помощью кадров-маячков о себе и о параметрах создаваемой сети. В соответствии с процедурой присоединения первый узел присоединяется непосредственно к PAN-координатору. Затем, если этот узел решает стать координатором (детали такого решения спецификация не оговаривает, но на практике это достигается конфигурированием узла), он также начинает распространять кадры-маячки о своей роли координатора, так что следующий узел может выбирать, к какому координатору он хочет присоединиться. Узел может присоединиться только к одному координатору: это гарантирует то, что в результате последовательных присоединений в сети образуется древовидная топология, пример которой показан на рис. 26.2.

PAN-координатор и координаторы сети могут маршрутизировать трафик в соответствии с топологическими связями дерева, при этом маршрут идет по кратчайшему пути на дереве связей, не обязательно заходя в PAN-координатор, — как это показано для пути между узлами 8 и 10 на рис. 26.2.

Спецификация 802.15.4 поддерживает также *ячеистую топологию* за счет связей между узлами-координаторами. Такие связи образуются поверх древовидной топологии с помощью протокола маршрутизации, который в беспроводных сетях способен самостоятельно создавать логические связи на основе физических. На рис. 26.2 показана одна связь ячеистой топологии: это связь между узлами 2 и 6. Однако сам протокол маршрутизации в спецификации 802.15.4 не определен.

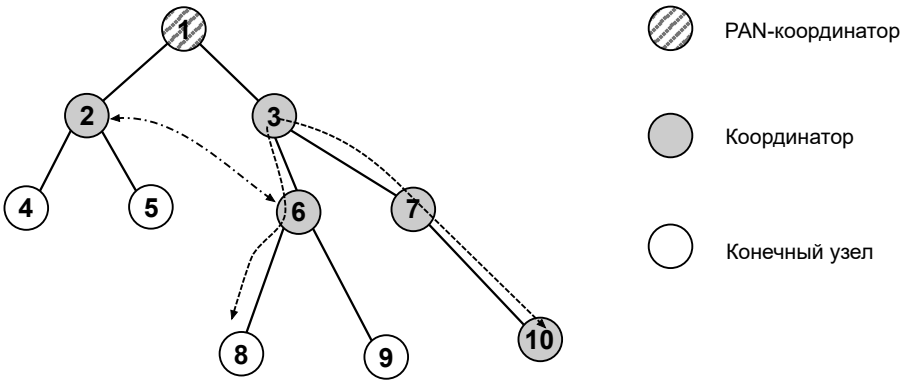


Рис. 26.2. Древовидная топология сети 802.15.4

На канальном уровне возможно использовать два алгоритма доступа к среде — распределенный алгоритм CSMA/CA и централизованный алгоритм CSMA/CA. При использовании *распределенного алгоритма CSMA/CA* все узлы сети, независимо от их роли, равноправно участвуют в соревновательном процессе доступа к среде в соответствии с правилами, изученными нами в главе 25.

Централизованный алгоритм CSMA/CA реализуется PAN-координатором сети и ее координаторами. Этот алгоритм основан на планировании доступа к среде на основе тайм-слотов и требует синхронизации работы узлов. В начале каждого периода планирования PAN-координатор передает в сеть кадр-маячок, который служит синхросигналом для узлов сети. Этот кадр говорит им, что за ним последует период, состоящий из 16 тайм-слотов, называемый суперкадром. Первые 8 тайм-слотов не закреплены за конкретными узлами, за их использование нужно соревноваться по правилам CSMA/CA. Оставшиеся 7 слотов (первый слот занят самим кадром-маячком) гарантированно выделяются нескольким конкретным узлам, очевидно, таким, у которых имеются приоритетные данные.

Решение о том, какой из двух алгоритмов доступа к среде будет использоваться, определяется конфигурацией PAN-координатора сети. Он объявляет о том, в каком режиме работает создаваемая им сеть, в своих кадрах-маячках.

Технологии ZigBee, Thread и Wi-SUN FAN

В результате адаптации и селекции тех или иных функций описанной выше платформы были созданы три технологии: ZigBee, Thread и Wi-SUN FAN, имеющие как общие черты, так и многочисленные отличия. Каждая из этих технологий имеет собственный стек протоколов, базирующийся на спецификации 802.15.4 (рис. 26.3).

В соответствии со спецификой областей применения (умным домом, промышленным цехом или системами контроля за локализованными объектами) разработчики технологии ZigBee и Thread выбрали из спецификации 802.15.4 такие параметры физического уровня, которые позволяют поддерживать довольно высокие скорости обмена данными — до 250 Кбит/с, но на небольших расстояниях между узлами в диапазоне 10–100 м.

Этими параметрами являются: частота 2,4 ГГц с шириной канала 2 МГц и модуляцией QPSK.

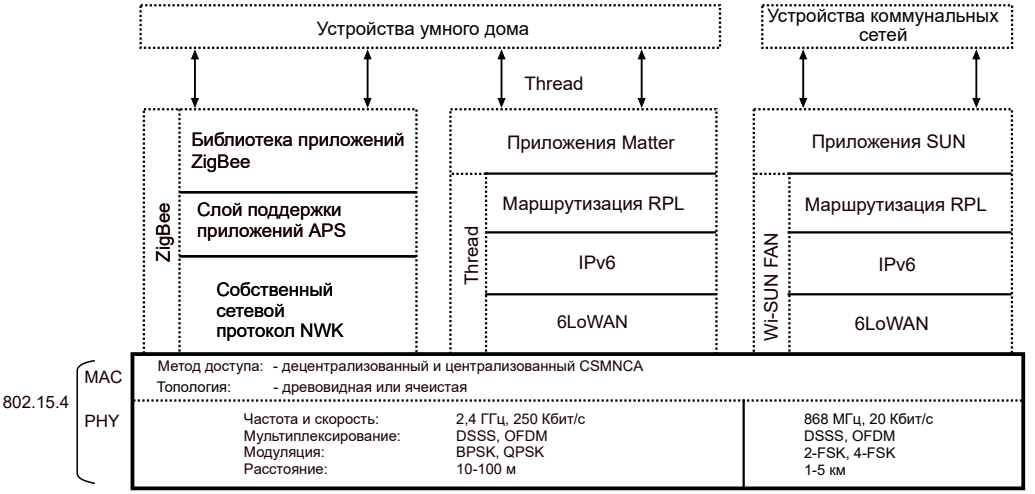


Рис. 26.3. Технологии, базирующиеся на спецификации 802.15.4

Технология Wi-SUN FAN, предназначенная для автоматизированного мониторинга и управления коммунальными сетями большой протяженности, использует другие параметры спецификации 802.15.4: частота 868 МГц с шириной канала 200 кГц и частотной модуляцией 2-FSK (2 значения частоты для кодирования одного бита) или 4-FSK (4 значения частоты для кодирования двух бит). Эти значения параметров уменьшили скорость передачи данных до 50–100 Кбит/с, но позволили увеличить расстояние между узлами до 1–5 км, что вполне соответствует потребностям коммунальных сетей. Увеличение расстояния произошло из-за комбинации трех факторов: низкая частота несущей (меньшее затухание), узкая полоса частот (меньший термальный шум на входе приемника — бóльшая чувствительность приемника), частотная модуляция (более устойчива к рассинхронизации приемника и передатчика и проявляется больше на больших расстояниях).

Стек протоколов технологии ZigBee

Разработчики **технологии ZigBee** пошли по пути создания своих собственных протоколов сетевого и прикладного уровней, а также библиотеки приложений, как это показано на рис. 26.3.

Протокол сетевого уровня NWK (NetWorK protocol) реализует функции обоих планов — плана пользовательских данных и плана управления. Этот протокол использует те же короткие 16-битные адреса узлов, которые были назначены узлу в процессе его присоединения к сети на MAC-уровне.

В ZigBee операция присвоения адресов делает больше, чем просто идентифицирует узлы, — она напрямую создает древовидную топологию сети. Для этого используется распределенная схема назначения адресов, которая работает следующим образом.

ПРИМЕЧАНИЕ

В каждой из трех технологий используются собственные названия для ролей узлов. Например, PAN-координатор в технологии ZigBee называется ZigBee-координатором, а технологии Thread — Thread Leader. В нашем тексте мы будем использовать уже описанные ранее термины спецификации 802. 15.4.

PAN-координатору выделяется диапазон адресов, достаточный для адресации всех узлов сети. Во время присоединения к нему координаторов он выделяет каждому из них поддиапазоны адресов. В свою очередь эти координаторы поступают аналогичным образом при присоединении узлов к ним, и так далее вниз по иерархии. При этом фиксируются отношения «родитель — потомок» между узлами сети. Получив поддиапазон адресов, координатор назначает один из них себе. В отличие от координаторов, конечный узел получает только один адрес.

Поскольку адреса иерархичны, становится возможным определить по адресу, к какому поддереву принадлежит узел.

Логика координатора при выборе пути для маршрутизации кадра заключается в следующем: если адрес назначения принадлежит моему поддиапазону адресов, то кадр нужно переслать вниз, тому потомку, поддиапазону которого принадлежит этот адрес; в противном случае кадр нужно переслать вверх к родительскому узлу. Решения о маршрутизации не требуют использования таблиц маршрутизации.

Для того чтобы такая схема работала, то есть не происходило истощения диапазонов адресов раньше времени, алгоритм распределенного назначения адресов регулируется параметрами — максимальным количеством дочерних узлов и максимальным количеством уровней дерева.

Ячеистая топология не может быть построена с помощью иерархии адресов, а следовательно, маршрут в сети не может быть найден по значению адреса назначения. Протокол NWK для построения ячеистой топологии использует традиционную маршрутизацию на основе таблиц. Работа протокола NWK в этом режиме аналогична работе протокола маршрутизации AODV, подробно описанного в главе 27.

На верхнем прикладном уровне в стеке протоколов ZigBee находится библиотека приложений, позволяющая быстро создавать приложения, управляющие устройствами ZigBee.

Между этим уровнем и сетевым уровнем находится протокол поддержки приложений (Application Support Sublayer, APS). Этот подуровень экранирует все транспортные детали сети ZigBee от прикладного уровня. Он выполняет функции протокола транспортного уровня: обеспечивает мультиплексирование сообщений между прикладными процессами и сетевым протоколом, а также их надежную доставку через сеть. Функции этого подуровня шире, чем функции транспортного протокола: он также создает логическую карту связей между устройствами сети, не привязанную к их сетевым адресам. Эта карта позволяет приложениям абстрагироваться на уровне устройства (например, в ней имеется связь «Переключатель управляет лампой»). На этом подуровне выполняются также функции безопасности, такие как шифрование и дешифрование.

Библиотека приложений состоит из программных модулей, описывающих устройства ZigBee и действия, которые над ними можно выполнять. Часть этих модулей устанавливается на самих устройствах.

Стек протоколов технологий Thread и Wi-SUN FAN

Разработчики **технологий Thread** и **Wi-SUN FAN** решили не создавать новый протокол сетевого уровня, а использовать стандартный протокол IPv6. Решение отдать предпочтение версии IPv6 в первую очередь связано с ее обширным пространством адресов по сравнению с версией IPv4, а это очень важно для интернета вещей с его многочисленными узлами. Такое решение позволяет использовать любые приложения по контролю и управлению разнообразными объектами, разработанные для стека TCP/IP с сетевым протоколом IPv6.

Для обеспечения работы протокола IPv6 поверх протоколов 802.15.4 необходимо было решить несколько инженерных задач и прежде всего:

- как передавать IPv6-пакеты размером от 1280 байт и более и с заголовком 40 байт и более в кадрах размером 127 байт;
- как маршрутизировать пакеты IPv6 в ячеистых топологиях сетей 802.15.4.

Для решения первой задачи был разработан **протокол адаптации 6LoWPAN**, который компрессирует заголовок пакета IPv6 до 2–5 байт, а заголовок UDP — до 4 байт, а также выполняет фрагментацию пакетов IPv6 в кадры 802.15.4 с последующей сборкой фрагментов в исходный пакет. Протокол адаптации 6LoWPAN экранирует от протокола IPv6 все особенности сети 802.15.4.

Проблема маршрутизации была решена с помощью **протокола RPL** (Routing Protocol for Low-Power and Lossy Networks — протокол маршрутизации для сетей с низким энергопотреблением и частыми потерями пакетов). Этот протокол маршрутизации ориентирован на сеть 802.15.4 с PAN-координатором, от которого строятся оптимальные маршруты к остальным узлам сети. Мы опять видим тут схожесть с решениями технологии Wi-Fi 802.11s, в которой протокол маршрутизации создает древовидные маршруты к узлу-порталу. Протокол RPL сконструирован так, что он не создает значительный служебный трафик в сети и тем самым экономит энергию узлов сети.

Стек протоколов технологии Thread заканчивается на уровне протокола маршрутизации RPL. Выше него находятся протокол транспортного уровня UDP и протоколы прикладного уровня CoAP и Matter. Matter является прикладным протоколом, ориентированным на мониторинг и управление умными «вещами», — он определяет модели устройств-«вещей» и команды управления ими, создавая среду для разработки приложений умного дома. Свои сообщения он упаковывает в сообщения протокола **CoAP** (Constrained Application Protocol) для передачи их через протоколы стека Thread по сети. Протокол CoAP представляет собой упрощенную версию протокола HTTP. Упрощения сделаны с учетом его работы на устройствах с ограниченными ресурсами.

Стек протоколов **технологии Wi-SUN FAN** во многом похож на стек Thread: он также заканчивается сетевым уровнем, на котором работают протоколы IPv6 и протокол маршрутизации RPL. Для разработки приложений сетей Wi-SUN FAN существует библиотека функций, использующая протокол CoAP. Основное отличие состоит в выборе параметров стека 802.15.4, позволяющих создавать протяженные сети мониторинга и управления датчиками и исполнительными механизмами коммунальных сетей.

Поддержка ячеистой топологии всеми тремя описанными технологиями является их большим преимуществом. Во-первых, с помощью ячеистой топологии можно расширить область покрытия сети. Во-вторых, ячеистая топология обладает повышенной надежностью по сравнению со звездообразной и древовидной топологиями.

Пример сети ZigBee

Рассмотрим сеть ZigBee, предназначенную для автоматизированного управления устройствами умного дома (рис. 26.4). В результате работы этой сети жители дома освобождаются от некоторых рутинных действий по обеспечению своего быта, таких как включение-выключение света, установка нужной температуры в доме и т. п.

Сеть состоит из 10 узлов: PAN-координатора, трех координаторов и шести конечных узлов. В сети создана ячеистая топология связей.

В этой сети в качестве узлов используются интеллектуальные бытовые устройства. Для поддержания сетевых функций ZigBee они оснащены встроенными миниатюрными контроллерами. Типичный контроллер выполнен как «система на микросхеме» (SoC, System on Circuit), то есть содержит внутри одной микросхемы все элементы контроллера — процессор, постоянную и оперативную память, радиоприемопередатчик. Типичный контроллер может иметь оперативную память 128 Кбайт и постоянную память в 1 Мбайт, потребляя при этом 1 микроампер тока в спящем состоянии и 30 миллиампер в режиме передачи данных. На этом контроллере работает программное обеспечение ZigBee — протоколы стека этой технологии и специализированное для определенного типа устройства приложение.

Центральным узлом сети, выполняющим роль *PAN-координатора*, в нашем примере является специализированный ZigBee-шлюз, который кроме функций ZigBee выполняет также функцию IP-шлюза интернета. Роль PAN-координатора могут играть и другие бытовые приборы и устройства, например интеллектуальные звуковые колонки. Важно, чтобы такое устройство питалось от электросети, так чтобы его контроллер мог всегда быть активным.

В приведенной на рис. 26.4 сети роль *координаторов* сети играют две лампочки и звуковая колонка. Постоянное питание от электросети позволяет контроллерам этих устройств все время находиться в активном состоянии, а их миниатюрное исполнение позволяет встраивать контроллеры даже в электрические лампочки.

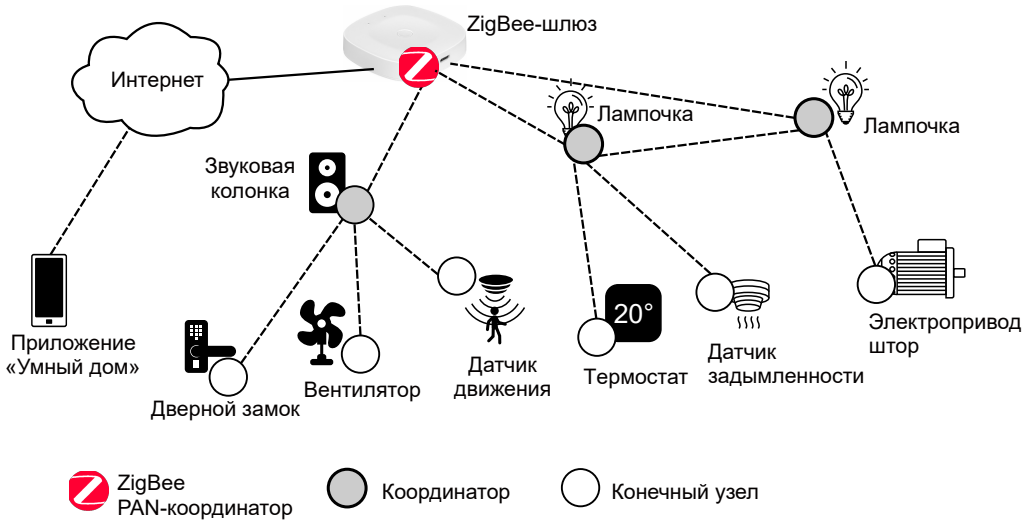


Рис. 26.4. Сеть умного дома ZigBee

На нижнем уровне сети находятся интеллектуальные устройства: датчик движения, дверной замок, термостат обогревательной системы дома, вентилятор, датчик задымленности и электропривод оконных штор.

Основным управляющим элементом этой сети является приложение «Умный дом», которое работает на компьютере или смартфоне (как в данном примере) и может взаимодействовать с ZigBee-координатором как локально, так и через интернет.

Сеть может работать по нескольким разным сценариям, которые соответствуют типичным ситуациям в жизни обитателей дома, таким как утро или вечер рабочего дня, отъезд в отпуск, переход от летнего режима к зимнему и др. Сеть ZigBee должна быть заранее сконфигурирована. В это конфигурирование входит задание конкретных параметров каждого типового сценария.

Пусть, например, выполняется сценарий «Вечер зимнего рабочего дня». При входе человека в дом срабатывает датчик движения, который и запускает дальнейшие действия. По команде от ZigBee-координатора включаются лампочки, а термостат переводит систему отопления в режим «18 градусов». ZigBee-контроллер передает также сообщение об опускании штор электроприводу, это сообщение маршрутизируется через лампочку. Аналогично срабатывают остальные умные вещи.

Технология LoRaWAN

Назначение и архитектура

Технология **LoRaWAN** (Long Range WAN) предназначена для устройств с очень низким энергопотреблением (срок службы до 10 лет на миниатюрной батарее), которым нужно поддерживать беспроводные связи с центральным узлом на больших расстояниях (10–15 км), при этом эти линии связи работают на сравнительно низких скоростях — от 1 Кбит/с до 20 Кбит/с. Одно из основных применений технологии LoRaWAN — сети датчиков, которые распределены по обширным областям и работают без обслуживания долгое время, периодически посылая данные измерений в центральный сервер. Технология LoRaWAN стандартизована организацией ITU-T в спецификации Y.4480.

Основным отличием технологии LoRaWAN от других технологий с низким энергопотреблением является метод модуляции с помощью так называемых **чирп-сигналов (chirp signal)** — сигналов с линейно возрастающей частотой. Chirp переводится как «чириканье» и отсылает к звукам, похожим на птичье чириканье, если воспроизвести такой сигнал в акустическом диапазоне. Чирп-модуляция и обеспечивает технологии LoRaWAN ее способность передавать данные малой мощности на большие расстояния. Этот метод модуляции получил имя **LoRa**, которое является и частью названия самой технологии.

Мы рассмотрим этот способ модуляции подробно далее, а пока ограничимся только тем фактом, что модуляция LoRa позволяет также мультиплексировать в одном частотном канале до семи независимых потоков данных *за счет выбора одного из параметров модуляции* (какого именно, мы узнаем немного позже, при рассмотрении модуляции LoRa).

Архитектура сети LoRaWAN показана на рис. 26.5. Беспроводная часть такой сети состоит из **конечных маломощных устройств** и **шлюзов**, обменивающихся данными через радио-

среду нелицензируемого частотного диапазона 868 МГц в Европе и 915 МГц в Северной Америке. В этих диапазонах технология LoRaWAN использует частотные каналы шириной 125 кГц, до 16 каналов в Европе и до 72 в Северной Америке.

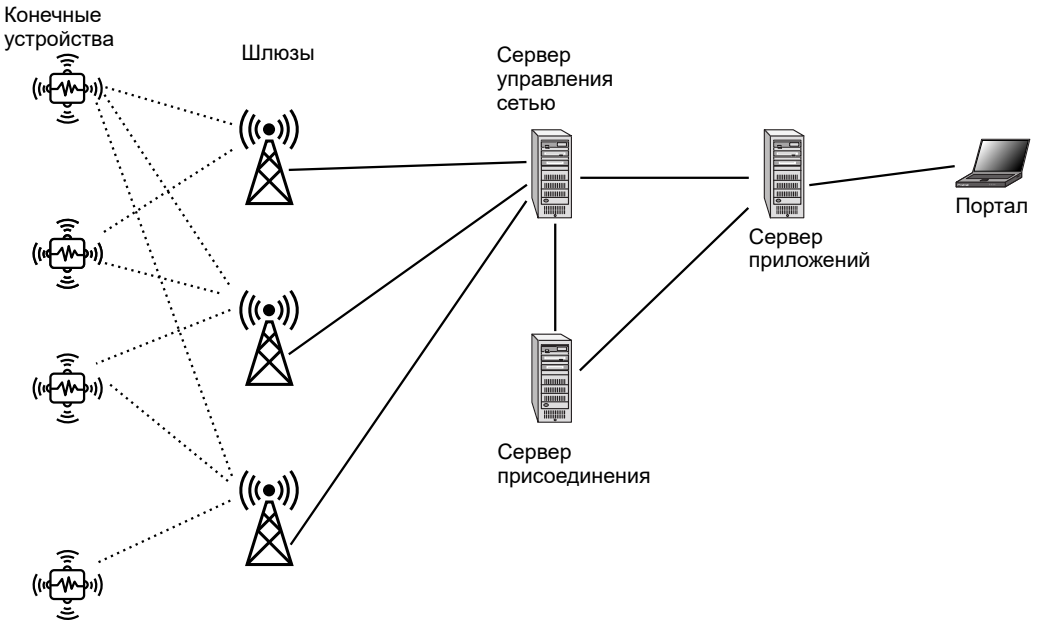


Рис. 26.5. Архитектура сети LoRaWAN

Конечные устройства выбирают для передачи очередного кадра данных один из частотных каналов случайным образом. При этом устройства работают с различными значениями параметра модуляции LoRa, что уменьшает вероятность возникновения коллизий, так как при наличии, например, 16 каналов и 7 независимых потоков в каждом канале сеть работает со 112 независимыми разделяемыми средами.

Тем не менее коллизии могут возникать, так как в качестве метода доступа к среде выбрана чистая ALOHA, метод, при котором устройство может начать передачу данных в любой момент времени без предварительного прослушивания среды и без случайных задержек метода доступа CSMA/CA, применяемом в сетях Wi-Fi.

Простота реализации метода доступа минимизирует количество операций, которое должно выполнить конечное устройство при передаче данных, а значит, сокращает энергопотребление конечного устройства по сравнению с вариантом CSMA/CA. Можно сказать, что разработчики технологии LoRaWAN сделали выбор в пользу сокращения энергопотребления и удешевления конечного устройства, пожертвовав эффективностью использования радиосреды, так как ALOHA позволяет использовать максимум 18 % полной пропускной способности среды из-за коллизий, тогда как эффективность метода CSMA/CA гораздо выше, до 60–70 %.

В сети имеется некоторое количество шлюзов, принимающих кадры от конечных устройств через радиосреду и передающих их далее специальному узлу сети, называемому сервером управления сетью.

Шлюз выполняет простую функцию *ретрансляции* полученного кадра: в отличие от точки доступа сети Wi-Fi или базовой станции сети LTE или 5G, радиосредой он не управляет. Шлюз состоит из двух блоков — приемопередатчика с модуляцией LoRa и блока конечного узла протокола IP. С помощью первого блока он обменивается потоками битов с конечными устройствами. Второй блок проверяет контрольную сумму полученного от конечного устройства кадра, и, если она корректна, то он переправляет этот кадр в виде дейтаграмм UDP/IP серверу управления сетью. Связь шлюза с сервером управления сетью может быть как проводная, так и беспроводная — в последнем случае, скорее всего, будет использоваться технология Wi-Fi для обеспечения необходимой скорости передачи данных потоков от многих конечных устройств.

В отличие от сотовых мобильных сетей, в сети LoRaWAN нет закрепления конечных устройств за определенным шлюзом: конечные устройства присоединяются к сети в целом, а не к отдельному шлюзу. Конечные устройства посылают свои кадры в разделяемую радиосреду, и *каждый* шлюз, который оказался в зоне покрытия данного конечного устройства, принимает этот кадр. Такой способ увеличивает вероятность того, что кадр будет принят сетью в случае мобильных конечных узлов, которые перемещаются между зонами покрытия отдельных шлюзов. В то же время этот способ прост в реализации, так как не требует применения процедуры эстафетной передачи при перемещении конечного устройства между зонами покрытия шлюзов — снова мы видим выбор в пользу простоты и, следовательно, уменьшения энергопотребления и стоимости узлов сети (конечных устройств и шлюзов).

Сервер управления сетью выполняет роль коммутатора сети. Он принимает кадры данных от шлюзов и переправляет их **серверу приложений** сети, на котором работают приложения, которые выполняют задачи обработки полученных от конечных устройств данных.

Так как посланный конечным устройством кадр может быть принятым несколькими шлюзами, то сервер управления сетью может получить несколько копий одного и того же кадра. Для того чтобы сервер мог понять, что это копии, конечное устройство помещает в заголовок кадра свой адрес и порядковый номер кадра. Сервер управления сетью выбирает из нескольких копий одну и отправляет ее северу приложений. Выбор осуществляется на основе качества сигнала, полученного шлюзом от конечного устройства: шлюз помещает параметры сигнала — отношение мощности сигнала к мощности помехи SNR и мощность сигнала P_{RX} — в заголовок кадра, передаваемого сетевому серверу, и сервер выбирает кадр с лучшим сочетанием этих параметров. Сервер управления сетью запоминает, от какого шлюза пришел этот кадр, чтобы передать ответ сервера приложений через тот же шлюз, который доложил о лучшем качестве сигнала.

Сеть LoRaWAN может иметь в своем составе *только один сервер* управления сетью: ее архитектура рассчитана на создание небольших сетей и плохо масштабируема. Поэтому в заголовках кадров LoRaWAN отсутствует адрес назначения, так как все кадры направлены одной точке назначения — единственному серверу управления сетью. Это позволяет сэкономить на размере заголовка, который имеет длину 7 байт (без опциональных расширений). Как и во многих беспроводных технологиях, разработчики технологии LoRaWAN

применили кадры небольшого размера, поэтому размер поля данных в кадре имеет значение от 30 до 200 байт (он зависит от скорости передачи данных, которая в свою очередь определяется параметром модуляции).

В заголовке кадра имеется один адрес — адрес устройства-отправителя, который имеет длину 4 байта, и, как и адрес станции Wi-Fi или же временный радиоидентификатор C-RNTI устройства пользователя сети LTE, назначается конечному устройству во время процедуры присоединения к сети на время его пребывания в сети. Устройства LoRaWAN также имеют 64-битные MAC-адреса, присваиваемые им на заводе-изготовителе, но в процедурах обмена информации они не используются.

Сервер управления сетью решает также задачу адаптивного управления скоростью конечных устройств, передавая им инструкции о смене скорости в зависимости от качества сигнала этого устройства, принимаемого шлюзом.

Сервер присоединения управляет процедурой присоединения конечного устройства к сети. Эту процедуру инициирует конечное устройство, которое посылает в радиосреду кадры запроса присоединения, указывая в нем свой полный MAC-адрес. Устройство также вычисляет так называемый **код целостности сообщения** (Message Integrity Code, **MIC**) для определенного слова-вызова и передает эту пару в запросе. У сервера присоединения хранится копия секретного ключа устройства, который он использует для вычисления своего значения MIC для переданного ему слова-вызова и, в случае совпадения со значением MIC в запросе, сервер считает, что конечное устройство успешно аутентифицировано.

В процессе аутентификации конечное устройство и сервер присоединения вырабатывают два секретных сессионных ключа — сетевой ключ и ключ приложений. Сервер присоединения передает сетевой ключ серверу управления сетью, а ключ приложений — серверу приложений. С помощью первого ключа шифруется трафик на участке «конечное устройство — сервер управления сетью», а с помощью второго — на участке «конечное устройство — сервер приложений». Первое шифрование предотвращает атаки в радиосреде и на шлюзы, которые могут располагаться в незащищенных местах. Второе шифрование обеспечивает защиту данных от вмешательства оператора связи, контролирующего сервер управления сетью.

Управление энергопотреблением

Конечные устройства сети LoRaWAN разбиты на три класса: класс A составляют устройства с минимальным энергопотреблением, класс C — с максимальным, а класс B занимает промежуточное положение.

Рассмотрим, каким образом экономят энергию своей батареи устройства класса A. Устройство этого класса проводит бóльшую часть времени в спящем состоянии, активизируясь только тогда, когда ему нужно передать очередной кадр через шлюз серверу приложений. После передачи кадра устройство снова засыпает, но после определенной паузы, называемой «задержка приема 1» просыпается, чтобы проверить, нет ли ответной передачи кадра от шлюза в его адрес (рис. 26.6). Устройство ожидает поступления кадра в течение периода времени, называемого «окно приема 1», при этом шлюзу дается время длительностью в 128 символов, чтобы начать передачу кадра. Если кадр в этот период не приходит, то устройство снова засыпает на период времени «задержка приема 2», после

чего снова проверяет приход кадра от шлюза. Если же и в этот раз кадр не приходит, то устройство считает, что ждать дальше бесполезно и больше не просыпается до того момента, когда ему нужно будет передать свой следующий кадр. Периоды задержки приема составляют от 1 до 15 секунд, длительность окна ожидания колеблется от 0,1 до 3 секунд.

Из диаграммы на рис. 26.6 видно, что основную часть времени устройство простаивает и тем самым экономит расход заряда своей батареи.



Рис. 26.6. Прием кадров устройством класса A

Класс A предназначен для таких устройств, как датчики, которые не очень часто генерируют данные и периодически отсылают их на сервер приложений, сами же данные не получают, но могут получить в принимаемых в окнах приема редких кадрах команды от приложения, например команду, изменяющую период измерений. Окном приема может также пользоваться сервер управления сетью, чтобы переслать квитанцию, подтверждающую прием кадра от конечного устройства.

Исполнительные устройства плохо соответствуют концепции устройств класса A, так как они должны принимать и отрабатывать команды от приложения в любой момент времени. Для таких типов устройств в технологии LoRaWAN имеется класс C. Эти устройства никогда не засыпают, но такой режим работы сокращает срок службы батарей, поэтому он больше подходит для устройств, питающихся от сети, например светофоров. Устройства класса B открывают окно приема периодически, а не только после передачи кадра.

Модуляция LoRa

Модуляция сигнала LoRa

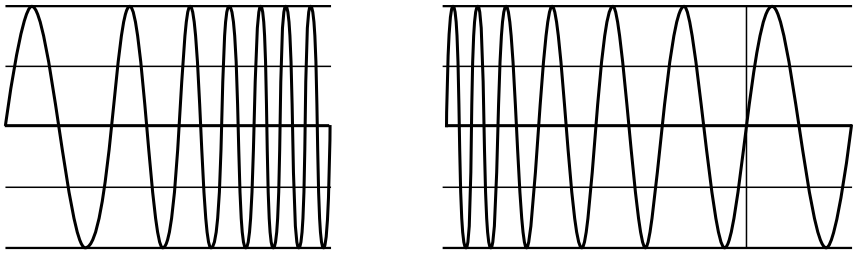
Мы опишем процессы модуляции и демодуляции LoRa весьма детально — настолько детально, насколько это позволяет жанр учебника по компьютерным сетям, а не по дискретной обработке сигналов. И сделаем это из уважения к уникальности этого метода модуляции, который специалисты из IEEE назвали «выдающимся» при описании его свойств на 34 страницах в своем учебном пособии по модуляции LoRa[1].

[1] Alireza Maleki, Ha H. Nguen, Ebrahim Bedeer, Robert Barton, A Tutorial on Chirp Spread Spectrum Modulation for LoRaWAN: Basics and Key Advances, IEEE ComSoc, VOLUME 5, 2024, pp. 4578-4612.

В методе модуляции LoRa в качестве несущего сигнала используется синусоидальный сигнал, у которого частота линейно изменяется с течением времени от некоторого минимального значения до некоторого максимального значения, равномерно заполняя полосу частот.

Этот тип модуляции известен довольно давно. Она применялась в военных радарах времен Второй мировой войны. Такая модуляция относится к классу линейной частотной модуляции (ЛЧМ), называемой также Chirp Spread Spectrum (CSS), дословно — расширение спектра чирп-сигналами. «Расширение спектра» в названии метода говорит о том, что спектр чирп-сигнала занимает всю полосу пропускания, отведенную для передачи данных, как и в других методах расширения спектра, рассмотренных ранее в главе 23.

На рис. 26.7 показана зависимость амплитуды чирп-сигнала от времени для двух вариантов сигнала — с возрастающей частотой (*а*) и с убывающей частотой (*б*).



а) Возрастающий чирп-сигнал *б*) Убывающий чирп-сигнал

Рис. 26.7. Амплитудное представление чирп-сигналов

На рис. 26.8 показана зависимость частоты от времени тех же чирп-сигналов, что и на рис. 26.7.

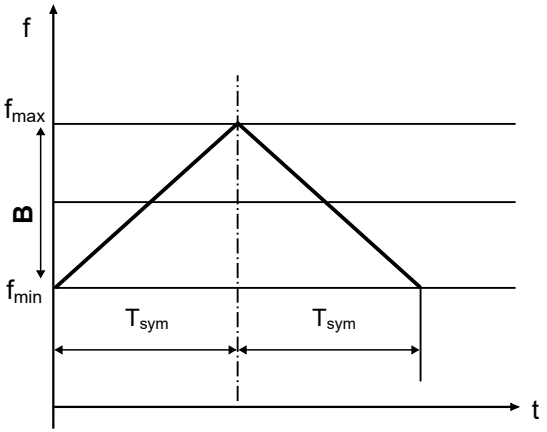


Рис. 26.8. Частотное представление чирп-сигналов

За период сигнала T_{sym} частота сигнала f изменяется от минимального значения f_{min} до максимального значения f_{max} по линейному закону:

$$f(t) = f_{min} + R \times t,$$

где R — скорость изменения частоты.

То, что частота сигнала изменяется в определенном диапазоне частот, от f_{min} до f_{max}, означает, что для передачи чирп-сигналов выделяется радиоканал с полосой пропускания $B = f_{min} - f_{max}$. При выбранной скорости изменения частоты R мы получаем длительность передачи отдельного сигнала S равной $T_{sym} = B/R$. За это время частота покрывает весь выделенный диапазон, будучи равномерно по нему распределенной.

Энергия такого чирп-сигнала почти равномерно распределена по частотам выделенного диапазона, это свойство и определяет главное достоинство чирп-модуляции — ее *низкую чувствительность к помехам*, так как сигнал устойчив не только к узкополосным помехам, как и все сигналы с расширенным спектром, но его заранее известная форма позволяет приемнику распознать сигнал даже при наличии широкополосных помех.

Рассмотрим теперь, каким образом можно использовать чирп-сигналы для кодирования и передачи двоичных данных, то есть каким образом можно модулировать такой специфический несущий сигнал. Наиболее простой способ модуляции подсказывает рис. 26.9 — возрастающий чирп-сигнал представляет, например, двоичную 1, а убывающий — двоичный ноль. На этом рисунке показано, как с помощью такого способа модуляции передается двоичная последовательность 10011.

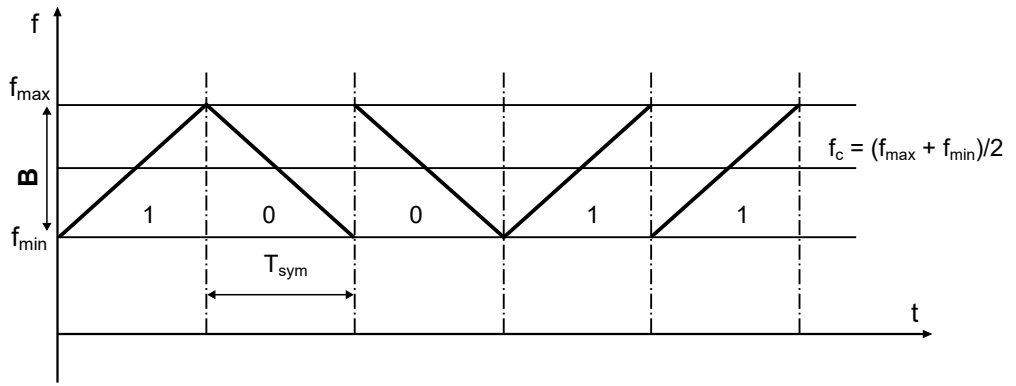


Рис. 26.9. Кодирование возрастающими и убывающими чирп-сигналами

При таком способе модуляции информационный сигнал имеет два состояния, то есть за время передачи одного символа передается один бит исходной информации. Несущей частотой сигнала f_c является средняя частота полосы пропускания, то есть $f_c = (f_{min} + f_{max})/2$.

Модуляция LoRa использует более эффективный подход: ее **чирп-сигнал имеет M состояний**, каждое из которых кодируется начальным сдвигом частоты сигнала относительно несущей (средней) частоты полосы частот.

Идея кодирования LoRa иллюстрируется рис. 26.10.

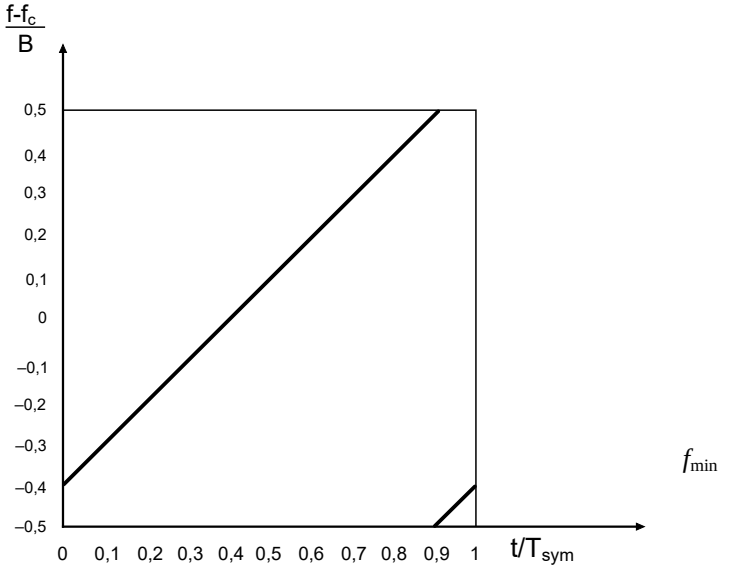


Рис. 26. 10. Символ кода LoRa

В начальный момент времени частота сигнала сдвинута на 0,4 от ширины полосы B вниз относительно частоты несущей f_c, то есть $f = f_c - 0{,}4B$ (шкала частот проградуирована в относительных единицах $(f - f_c)/B$). Далее чирп-сигнал нарастает со скоростью R до тех пор, пока не дойдет до верхней границы полосы частот f_{max}. После этого частота сигнала скачком уменьшается до нижней границы полосы f_{min} и продолжает расти до конца периода передачи данного символа до значения $f_c - 0{,}4\ B$, с которой началась передача данного символа. То есть за время передачи символа сигнал LoRa изменяется с линейной скоростью R, пробегая все частоты из отведенной полосы частот, но из-за начального смещения делает это за два приема.

Для n-го символа кода справедливо следующее выражение начального сдвига частоты:

$$s(n) = -B/2 + n \times B/2,$$

где $n = (1, 2, ..., M)$.

Модулятор LoRa генерирует чирп-сигналы цифровым способом: он представляет чирп-сигнал в виде последовательности дискретных во времени значений частоты, которые затем преобразуются аналого-цифровым преобразователем в аналоговые. Частота дискретизации выбрана равной ширине полосы частот B. Как мы знаем из теоремы Котельникова — Найквиста, для точного воспроизведения функции необходимо, чтобы частота дискретизации была в два раза больше, чем ширина полосы частот этой функции. Однако специальная форма чирп-сигнала с предсказуемой линейной зависимостью частоты от времени ослабляет требования к частоте дискретизации.

На каждом шаге дискретизации частота увеличивается на величину B/M, чтобы за M шагов сигнал покрыл всю полосу частот B (рис. 26.11). При этом скорость роста частоты R равна:

$$R = \Delta f/\ \Delta t = B \times B/M = B^2/M.$$

В модуляции LoRa значение M всегда равно степени 2 по формуле:

$$M = 2^{SF}, \text{ где } SF = \{7, 8, 9, 10, 11, 12\},$$

поэтому M принимает значения {128, 256, 512, 1024, 2048, 4096}.

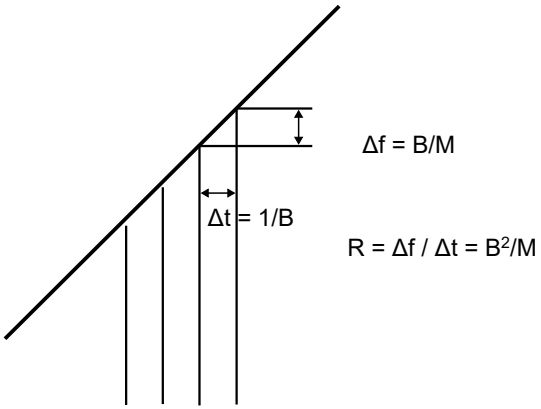


Рис. 26.11. Дискретное наращивание частоты сигнала LoRa

Параметр SF (Spread Factor) называется *фактором расширения*. Он расширяет сигнал во времени: чем больше этот фактор, тем более продолжительным является время передачи одного символа T_{sym}, так как:

$$T_{sym} = B/R = M/B = 2^{SF}/B,$$

или же можно записать

$$T_{sym} = 2^{SF} \times \Delta t,$$

где $\Delta t = 1/B$ — интервал дискретизации.

Другими словами, символ генерируется из M шагов, а длительность каждого шага фиксирована — это длительность интервала дискретизации. Чем больше значение SF, тем больше M, а значит, больше длительность символа, то есть фактор SF действительно расширяет сигнал во времени.

Нетрудно видеть, что значение SF определяет количество бит, передаваемых одним символом кода LoRa, так как $\log_2 M = \log_2 2^{SF} = SF$.

Например, если мы выбрали SF равным 9, то один символ кода передает информацию 9 исходных бит, при этом количество символов в коде $M = 512$ и номер символа n принимает значения от 0 до 511.

При полосе канала 250 кГц в диапазоне 868,0–868,25 МГц мы получим смещение между символами $250 \times 10^3/512 = 1024$ Гц. Это означает, что первый символ 9-битного кода ($n = 0$) будет иметь начальную частоту 868,0 МГц, второй — 868,001024 МГц, третий — 868,002048 МГц, а 512-й — 868,25 МГц. Длительность символа этого кода составляет $2^9/250 \times 10^3 = 0{,}002048$ с = 2,048 мс.

Теперь рассмотрим способы реализации модулятора и демодулятора сигналов LoRa.

Модуляция сигнала LoRa выполняется квадратурным модулятором, работа которого описана в главе 7. Для приведения случая частотной модуляции к фазовой, с которой работает квадратурный модулятор, необходимо воспользоваться определением мгновенной скорости. Это определение говорит, что мгновенная скорость равна производной зависимости фазы от времени.

Поэтому, проинтегрировав линейную зависимость мгновенной частоты от времени сигнала LoRa, то есть зависимость $f(t) = f_{min} + R \times t$, мы получим выражение для мгновенной фазы этого сигнала, которая нужна для того, чтобы воспроизвести сигнал LoRa с помощью квадратурного генератора.

Опуская промежуточные преобразования и выражая скорость изменения частоты через ширину полосы B и фактор SF, получаем выражение для мгновенной фазы сигнала LoRa $\varphi(t)$:

$$\varphi(t) = 2\pi\,[-B/2 \times t + (B^2/2^{SF}) \times t^2].$$

Как уже было сказано, квадратурный модулятор LoRa работает в дискретном времени, поэтому эта формула преобразуется в набор значений фазы для M моментов времени и с помощью этих значений вычисляются значения синфазной и квадратурной составляющих модулирующего сигнала, то есть $i(t_i) = A\cos(\varphi(t_i)$ и $q(t_i) = A\sin(\varphi(t_i)$, где индекс i изменяется от 0 до $(M-1)$.

Для ускорения операции модуляции дискретные значения синфазной и квадратурной составляющих вычисляются заранее для каждого из семи значений SF и помещаются в память модулятора. Поэтому при необходимости генерации сигнала для определенного значения SF из памяти модулятора с нужной частотой извлекаются заранее вычисленная последовательность значений синфазной и квадратурной составляющих, соответствующих данному значению SF. Затем с помощью цифро-аналоговых преобразователей дискретные значения синфазной и квадратурной составляющих преобразуются в аналоговые сигналы и в аналоговой форме перемножаются на синусоиды несущего сигнала частоты f_c, то есть работает стандартная схема квадратурного модулятора.

Демодуляция сигнала LoRa требует точной синхронизации приемника с передатчиком, поэтому в начале передачи каждого символа передается преамбула из нескольких «простых» немодулированных чирп-сигналов, у которых начальная частота не сдвигается (как у сигнала на рис. 26.7). После достижения синхронизации приемник выполняет выделение синфазной и квадратурной составляющих сигнала, как это было описано в главе 7. На этом аналоговая часть демодуляции заканчивается, синфазный и квадратурный сигналы оцифровываются и дальнейшие преобразования выполняются над дискретными значениями этих двух составляющих сигнала.

Для нахождения начального сдвига частоты сигнала (то есть для нахождения значения принятого символа) полученные значения чирп-сигнала перемножаются с эталонным нисходящим чирп-сигналом. Этот процесс иллюстрируется рис. 26.12. Умножение выполняется над значениями сигналов как комплексных чисел, то есть, например, исходный чирп-сигнал представляется в виде последовательности значений комплексных чисел $r(n)$, где $r(n) = i(n) + j\,q(n)$, таким же образом представляется и эталонный чирп-сигнал.

Перемножение двух чирп-сигналов, изменяющихся в противоположных направлениях с одинаковой скоростью, приводит к удалению переменной частоты и оставляет синусо-

идальный сигнал постоянной частоты. Частота результирующего монотонного сигнала пропорциональна начальному сдвигу частоты исходного сигнала, то есть ее значение и есть результат демодуляции.

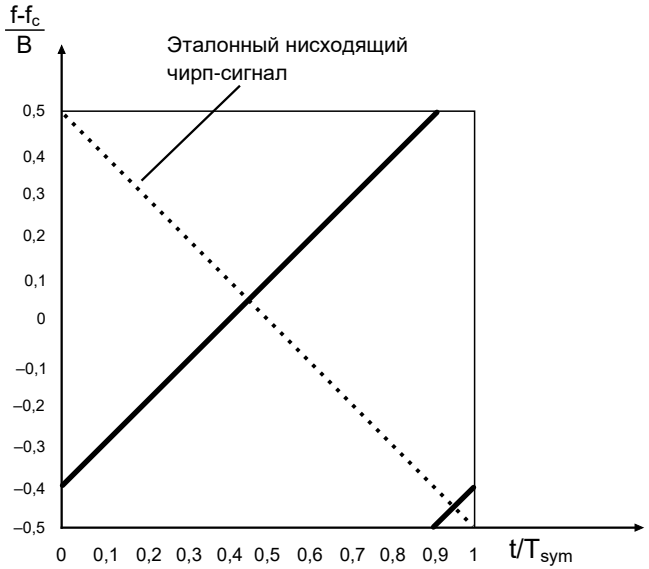


Рис. 26.12. Умножение принятого восходящего чирп-сигнала на эталонный нисходящий чирп-сигнал

Частоту монотонного сигнала находят с помощью дискретного преобразования Фурье результата произведения чирп-сигналов. Это преобразование дает на выходе набор значений, состоящий из *M* частот, и *частота с наибольшей амплитудой* соответствует смещению частоты исходного символа, то есть значению передаваемого двоичного кода, состоящего из SF бит. На графике эта частота заметно выделяется среди набора частот разложения Фурье своей амплитудой. Процесс демодуляции на этом завершен.

Уникальность модуляции LoRa проявляется в том, что она оказалась очень *устойчивой к помехам*. Приемник LoRa может устойчиво распознавать чирп-сигналы даже при отношении уровня сигнала к уровню помех (SNR) в –20 дБ, то есть в том случае, когда *уровень сигнала в 100 раз меньше, чем уровень помех*[1]. Другие схемы модуляции требуют, чтобы уровень сигнала превышал уровень помех, то есть чтобы значение SNR было положительным. Например, для такого метода модуляции, как 2-FSK, который считается весьма устойчивым к помехам, уровень SNR должен быть выше 5–10 дБ. Соответственно, беспроводные линии связи, на которых используется модуляция LoRa, могут иметь гораздо больший бюджет (то есть допускать гораздо большее ослабление сигнала), на 20–30 дБ больше, чем при других методах модуляции. Поэтому радиус действия беспроводных линий с модуляцией LoRa оказывается таким большим, до 10–15 км.

[1] Fabio Busacca, Stefano Mangione, Sergio Palazzo, Francesco Restuccia, Ilenia Tinnirello. SDR-LoRa, an open-source, full-fledged implementation of LoRa on Software-Defined-Radios: Design and potential exploitation. Computer Networks, Volume 24 1 , March 2024.

Способность подавлять помехи проявляется еще в одном полезном свойстве модуляции LoRa — ортогональности ее сигналов при различных значениях фактора SF. Другими словами, приемник, настроенный на распознавание сигналов с определенным значением SF, подавляет сигналы с другими значениями SF точно так же, как он подавляет помехи. А это означает, что в одном частотном канале могут сосуществовать без коллизий сигналы с различными значениями SF. Именно этот параметр модуляции имелся в виду, когда было сказано, что в частотном диапазоне с 16 каналами могут одновременно работать без коллизий $16 \times 7 = 112$ приемопередатчиков.

Сетевая технология LoRaWAN рассчитана на объединение в сеть большого числа устройств, обменивающихся данными только с центральным узлом, но не между собой. В этом она похожа на сети BSS технологии Wi-Fi или же сотовые мобильные сети. В то же время имеется потребность в одноранговых сетях с узлами низкого энергопотребления, но большого радиуса действия, например в сетях дронов. Поэтому имеется ряд работ, в которых такая организация сети с использованием модуляции LoRa исследуется.

ГЛАВА 27 Сети дронов

Области применения и классификация дронов

Беспилотные летательные аппараты (Unmanned Aerial Vehicles, **UAV**), называемые также **БПЛА** и **дронами**, — это устройства, летающие автономно или под управлением с земли. Они обладают широко признанной репутацией современного военного средства. Однако концепция использования в военном деле беспилотных летающих устройств существует уже несколько сотен лет. Более того, в далеком прошлом существовало много примеров практического использования этих средств как для атакующих действий, так и для разведывательных целей: например, беспилотные аэростаты использовались для нанесения ударов по осажденной Венеции в 1849 году, а во время испано-американской войны 1898 года дистанционно управляемые фотокамеры устанавливались на воздушных змеях.

Область БПЛА стремительно развивается и расширяется за счет спектра гражданских применений, когда требуется работать в труднодоступных или опасных районах, где опасность для людей существует не только на земле, но и на борту пилотируемого средства. Во многих случаях дроны применяются тогда, когда такое решение проще и дешевле, чем использование пилотируемых средств. И наконец, в некоторых случаях беспилотные летающие средства являются единственным возможным решением. Так, в регионе, подвергшемся катастрофе, дроны, выполняя роль зависших над землей базовых станций, позволяют быстро восстановить сотовую связь или доступ к интернету. Видеокамеры дронов оперативно передают в центр изображение зоны бедствия, а также критические данные об обстановке, например информацию о направлении распространения огня, что позволяет существенно сократить время реагирования пожарных команд, работающих на земле.

Особенно перспективным направлением использования дронов является скоординированная работа группы дронов, объединенных в сеть, или рой, дронов. Такая работа требует решения комплекса коммуникационных задач, специфических для компьютерных сетей.

ПРИМЕЧАНИЕ

Использование дронов регулируется законодательно. Воздушное пространство делится на две категории: контролируемое воздушное пространство, для входа в которое каждое воздушное судно, в том числе и БПЛА, должно получить официальное разрешение, и неконтролируемое воздушное пространство, где разрешения получать не нужно.

В то время как об узлах классических компьютерных сетей — компьютерах — мы знаем довольно много, дроны являются для специалистов по сетевым технологиям гораздо менее знакомым типом коммуникационного узла, поэтому имеет смысл начать изучение этого типа сетей с краткой классификации дронов.

Существует большое количество параметров, по которым можно классифицировать дроны: область использования, габариты и вес, дальность, скорость и максимальное время пребывания в полете, грузоподъемность, стоимость и др. Не претендуя на исчерпывающую таксономию, мы остановимся на нескольких важных характеристиках БПЛА.

Фундаментальное различие БПЛА проистекает из *аэродинамических принципов*, лежащих в основе того или иного летательного аппарата. Это может быть:

- **Винтокрылая система**: квадрокоптеры, гексакоптеры, октокоптеры. Они могут зависать в воздухе, взлетать и приземляться вертикально.
- **БПЛА с фиксированным крылом.** Они используют крылья самолетного типа для создания подъемной силы; требуют наличия взлетно-посадочной полосы; более энергоэффективны для длительных полетов.
- **Гибридные БПЛА** — сочетание винтокрылой системы с фиксированным крылом, что дает дрону возможность вертикального взлета с последующим крейсерским полетом.

К этому примыкает классификация *по способу запуска и посадки* дронов. Помимо уже упомянутых винтокрылых дронов, не требующих взлетно-посадочной полосы, и аппаратов с фиксированным крылом, нуждающихся в таковой, используются также тактические дроны с катапультами/пусковыми установками и небольшие полевые дроны с ручным запуском/посадкой на парашюте.

По весу и габаритам дроны можно разделить на следующие категории:

- Миниатюрные БПЛА — вес менее 250 граммов, для использования в помещениях в сфере научных исследований и образования.
- Мини-БПЛА весом до 25 кг, габариты которых позволяют перевозить военные версии этих аппаратов в рюкзаках солдат. Характеризуются коротким временем (5–30 минут) и малой высотой (менее 330 м) полета, используются для гражданской фотосъемки, картографии.
- Малые БПЛА (25–150 кг) для инспектирования промышленных объектов.
- Тактические БПЛА (150–600 кг) для наблюдения за полем боя.
- Средневысотные БПЛА (Medium Altitude Long Endurance, MALE) с большой продолжительностью полета, весящие до 1500 кг, используются для военных разведывательно-рекогносцировочных целей (например, MQ-9 Reaper) и работают на высоте до 9000 м в полетах, длящихся много часов, на расстояния сотен километров.
- БПЛА большого радиуса действия (High Altitude Long Endurance, HALE) весят более 1500 кг. Этот тип БПЛА предназначен для выполнения задач, требующих длительного (несколько часов и даже дней) наблюдения, сбора и передачи данных. Это самые большие и сложные БПЛА.

По максимально достижимой высоте дроны можно разделить на малые, средние и большие. Максимальная высота полета малых дронов составляет менее 300 м, а больших — более 5500 м. Высота полета средних дронов находится между этими диапазонами.

Максимальная дальность полета БПЛА варьируется от нескольких километров для любительских дронов до нескольких десятков километров для дронов ближнего радиуса действия. БПЛА средней дальности летают на расстояния до 250 км. БПЛА длительного действия преодолевают 1000 км и более.

Особенно большой дрон используется в технологии 5G NTN (см. раздел «Прямая связь с телефоном» главы 24) в качестве летающей базовой станции (High Altitude Platform Station, HAPS). Дрон HAPS имеет размах крыльев 25–80 м, что сравнимо с небольшим пассажирским самолетом, летает на высоте 18–22 км, проводя в полете от нескольких недель до нескольких месяцев.

Классификация БПЛА *по степени автономности от команд оператора*:

- Минимальной автономностью обладают летающие устройства, которые не имеют автопилота и полностью вручную управляются оператором с земли.
- Полуавтоматические БПЛА. Здесь дрон оснащен средствами автопилота для автоматического поддержания горизонтального и вертикального положения. Для этого встроенный GPS-приемник и барометр непрерывно отслеживают текущие координаты дрона, при этом полетный контроллер сравнивает текущее положение дрона с заданным. Если дрон смещается под действием ветра или дрейфа, небольшая коррекция скорости двигателя возвращает его в исходное положение. Полуавтоматический дрон, используя навигационные датчики, способен автоматически следовать по заданному маршруту, составленному из точек маршрута (координат GPS), которые оператор загружает перед полетом. При необходимости оператор может перехватить управление. Такой тип дрона идеально подходит для картографирования, геодезии или поисковых миссий, где необходимо многократно и точно выполнять один и тот же маршрут.
- Автономные дроны — заранее запрограммированные устройства, способные самостоятельно следовать в заданный район по заданному маршруту и выполнить поставленную задачу без инструкций оператора.

Сети с использованием дронов и сети на основе дронов

Термины **«сеть дронов»** и **«рой дронов»** часто используются взаимозаменяемо, но иногда термину «рой дронов» придается более узкое значение. Рой дронов — это такой тип сети дронов, для которой характерна высокая степень координации как интеллектуальных действий, так и физических перемещений в пространстве — полет строем, предотвращение столкновений, групповая навигация. Рой дронов часто использует алгоритмы, вдохновленные биологическими стаями и роями (птицами, пчелами, рыбами). Например, подобно птицам в стае, дроны адаптируют свое поведение в зависимости от поведения соседних узлов для достижения групповых целей, стремясь действовать как единое целое. Далее оба эти термина будут использоваться как синонимы.

Во многих случаях эффективность выполнения миссии сетью дронов оказывается значительно выше, чем работа отдельного БПЛА. Сравнение отдельного дрона с сетью дронов весьма условно, так как многое зависит от того, какие задачи перед ними поставлены. Одиночный БПЛА, как правило, предназначен для выполнения более простых, изолированных задач типа аэрофотосъемки или точечного сбора данных. Перед сетью БПЛА ставятся более сложные задачи, включающие самоорганизацию и распределенное принятие решений, например реагирование на стихийные бедствия, выполнение поисковых операций.

По сравнению с системой из одного дрона рой или сеть из нескольких дронов имеет следующие преимущества:

- *Масштабируемость* — в то время как использование более крупного дрона в системе с одним дроном может ненамного увеличить дальность действия, системы с несколькими БПЛА могут расширить зону покрытия существенно.
- *Живучесть* — сбитие одного БПЛА во время выполнения задания в системе с одним дроном считается отказом. Однако для систем с несколькими дронами выход одного дрона из-под контроля не представляет серьезной угрозы, поскольку другие БПЛА продолжат работу.
- *Скорость выполнения задания* — при использовании систем с несколькими дронами скорость вычислений выше, поскольку БПЛА могут обрабатывать задачи параллельно.
- *Автономность* — обычно в системах с одним БПЛА оператор непосредственно управляет всеми системами БПЛА в режиме реального времени, в то время как в большинстве систем с несколькими БПЛА дроны чаще полагаются на бортовую автоматику.

Итак, наличие нескольких дронов в сети обеспечивает большее покрытие, стабильность, меньшую задержку и гибкость выполнения миссии, в то время как схема с одним дроном гораздо более уязвима и ограниченна.

С другой стороны, система, состоящая из одного дрона, намного проще. Для отдельного дрона не требуется решать задачу координации его движения с другими дронами. Как правило, он движется по заранее запрограммированной траектории, в то время как перемещение дронов, связанных в сеть, является результатом самоорганизации, часто с использованием искусственного интеллекта. Большая сложность сети дронов приводит к увеличению накладных расходов на организацию связи и энергообеспечение.

Сети дронов имеют много различных вариаций в зависимости от архитектуры, специализации, топологии и других характеристик. Все множество сетей дронов может быть разделено на два больших класса — сети, в которых дроны играют вспомогательную роль, и сети, в которых они являются основными, базовыми компонентами.

Беспроводные сети с использованием дронов (UAV-Assisted Wireless Networks) — это *наземные* беспроводные сети, в которых БПЛА используются в качестве *вспомогательных* средств. Обычно основная сеть уже существует, а дроны присоединяются к ней по запросу, чтобы временно улучшить ее инфраструктуру. Дрон может выступать в качестве:

- *Воздушного ретранслятора* — дрон в такой сети является промежуточным звеном между пользователями и базовыми станциями. Таким образом он позволяет пользователям передавать данные на базовые станции даже при отсутствии прямой видимости или при заблокированных каналах связи.
- *Летающей базовой станции* — дрон обеспечивает прямой доступ к наземным пользователям и тем самым помогает временно восстановить зону покрытия сотовой сети после катастрофы или разгрузить трафик в густонаселенных городских районах. Дроны используются для быстрого развертывания беспроводной коммуникационной инфраструктуры, например, в области, где отсутствует покрытие сигналами мобильной сотовой связи, но она временно нужна для поисковой операции. В этой ситуации требуются неподвижные зависающие дроны. Зависший в небе дрон играет роль сотовой вышки или выделенного коммуникационного сервера, предоставляя своим клиентам на земле доступ к интернету, к услугам голосовой и видеосвязи.

Беспроводные сети на основе дронов (UAV-Supported Wireless Networks) — это беспроводные сети, в которых БПЛА являются *неотъемлемой частью*, образуют основную

коммуникационную инфраструктуру и являются конечными узлами сети. Именно к этому классу относятся **летающие самоорганизующиеся сети** (Flying AdHoc Network, **FANET**), которые находятся в фокусе нашего внимания в этой главе.

ПРИМЕЧАНИЕ

Сети дронов часто рассматривают как подмножество интернета вещей, где умными «вещами» являются дроны. Действительно, дроны оснащены датчиками, процессорными блоками и коммуникационными модулями, для управления и обработки данных сети дронов подключены к облачным или периферийным системам. Вес это стандартные компоненты интернета вещей. Вместе с тем дроны обладают очень высокой мобильностью и перемещаются в трехмерном пространстве, что не свойственно большинству объектов интернета вещей. Навигация в реальном времени, предотвращение столкновений, синхронизация полета с другими дронами - все эти и другие специфические свойства выделяют сети дронов из мира интернета вещей в особый класс.

Архитектура связей дронов

Архитектура сети дронов может быть отнесена к одному из трех определенных ниже типов, каждый из которых будет рассмотрен более подробно далее в этом разделе.

- **Централизованные сети дронов**, когда один центральный объект (наземная станция или ведущий БПЛА) контролирует маршрутизацию, координацию и агрегацию данных. Для таких сетей характерна звездная топология, центральной точкой которой является центральный БПЛА или наземная станция. Как и все централизованные системы, данная архитектура характеризуется плохой масштабируемостью и низкой отказоустойчивостью. К числу преимуществ относится простота управления.
- **Децентрализованные (распределенные) сети дронов**, в которых каждый дрон самостоятельно принимает независимые сетевые решения, поскольку отсутствует фиксированный управляющий блок. Это, с одной стороны, усложняет координацию дронов, а с другой стороны, делает систему более устойчивой и масштабируемой.
- **Иерархические сети дронов** имеют многоуровневую, часто кластерную, структуру, образованную дронами разных типов — ведущие дроны, ретрансляционные дроны, рабочие дроны. Эта схема представляет собой компромисс между двумя рассмотренными выше архитектурами.

Схема управления изолированным дроном

На рис. 27.1 показан дрон, соединенный с базовой станцией мобильной сотовой связи, через которую он получает команды от оператора, находящегося в наземном командном центре, и передает туда собранные им данные.

Для того чтобы дрон мог пользоваться услугами связи некоторой мобильной сотовой сети, например сети 4G/LTE, радиоблок дрона должен поддерживать стандарты этой сети и иметь в своем составе SIM-карту, зарегистрированную у провайдера этой сети.

В определенной фазе полета (например, после набора определенной высоты или прохода некоторой точки на местности) дрон включает свой радиоблок. Радиоблок выполняет поиск синхросигналов базовой станции своей сети и, если находит их, то выполняет стандартную процедуру присоединения к сети, описанную для сети LTE в главе 24. Если

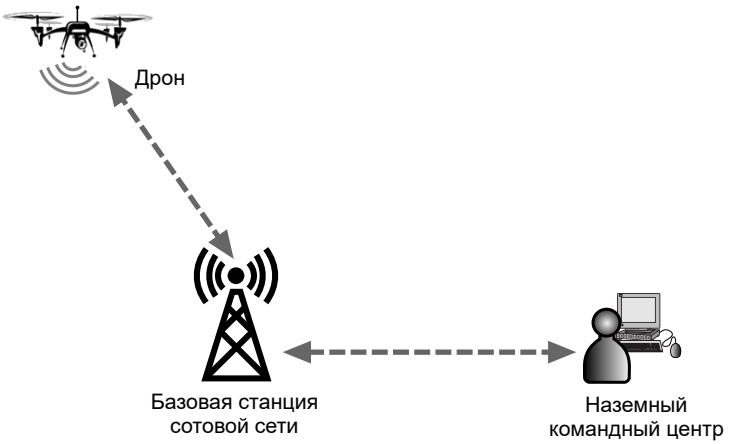


Рис. 27.1. Простейшая схема управления дроном

присоединение проходит успешно, то с этого момента дрон может пользоваться услугами этой сети, выходить через нее в интернет, обмениваться через интернет данными с наземным командным центром и получать от него инструкции по полету, если они необходимы. В этой схеме дрон является *изолированным*, так как не связан с другими дронами. В отношении зависимости дрона от команд оператора могут быть различные сценарии: от полностью ручного управления до полной автономии.

Данная схема является централизованной и может быть расширена на несколько изолированных дронов, каждый из которых независимо от других обменивается данными и командами с наземным центром через общую базовую станцию.

Связь дрона с наземным центром может осуществляться с использованием не только сотовой связи, но и других беспроводных технологий доступа, таких как спутниковая связь, Wi-Fi или другая прямая радиосвязь.

Централизованная сеть дронов

Рассмотрим схему, когда к одной базовой станции подключается несколько дронов. Как и в первом случае, здесь каждый дрон связан напрямую с базовой станцией. Базовая станция играет роль центрального элемента инфраструктуры. Она передает данные и инструкции оператора каждому дрону. Но сверх того, как показано на рис. 27.2, базовая станция предоставляет дронам возможность обмениваться сообщениями друг с другом (рисунок иллюстрирует связь дронов *A* и *B*), а значит, они уже не являются изолированными узлами, а образуют *сеть дронов,* или *рой дронов*, в котором базовая станция выполняет функцию маршрутизатора.

Топология централизованной сети (рис. 27.2) — «звезда», в которой центральным элементом является узел инфраструктуры доступа, в данном примере базовая станция сотовой сети. Этот вариант ввиду своей простоты является практичным решением, особенно когда количество одновременно подключенных дронов мало, зона покрытия небольшая и задачи миссии несложные. Например, такая схема могла бы использоваться при наблюдении за большим скоплением людей во время масштабных публичных мероприятий. Поскольку

дроны контролируют место события с неба, ожидается, что они смогут легко и быстро обнаружить любое подозрительное движение, в ответ на которое смогут вмешаться агенты безопасности.

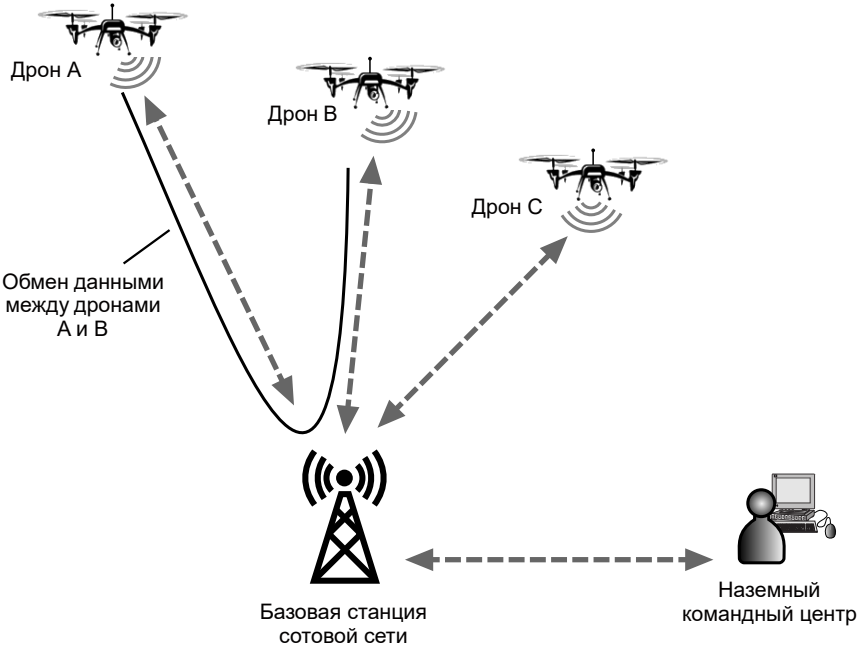


Рис. 27.2. Централизованная архитектура связи дронов

Однако у такой схемы есть принципиальные ограничения:

- Даже если дроны *A* и *B* находятся близко друг от друга, они не могут связываться друг с другом напрямую. Их сообщениям приходится путешествовать по существенно более длинному пути через центральный узел данной схемы, а значит, увеличивается задержка обмена данными.
- Как и все централизованные системы, эта схема обладает *единой точкой отказа* — отказ базовой станции является критическим для работоспособности данной схемы.
- Зона действия дронов ограничена дальностью связи между дроном и базовой станцией.

Для увеличения зоны действия можно использовать *спутниковую связь*, поскольку у спутника, или тем более у сети спутников, зона покрытия значительно выше. Так же, как и в предыдущем случае, дроны могут общаться через спутник с наземным командным пунктом, а также между собой (на рис. 27.3 показана связь дронов *A* и *B*).

Однако тяжелое, энергоемкое и дорогое оборудование для связи со спутником предъявляет более высокие требования к характеристикам дрона и делает в целом такое решение более дорогим. Помимо этого спутники вносят дополнительную задержку в передачу телеметрических и командных сигналов. Например, задержка низкоорбитальных спутников Starlink составляет до 20–40 мс, а геостационарных — 500–700 мс. Эта задержка должна учитываться в работе автопилота и в других алгоритмах управления полетом. Из-за задержки

сигнала со спутника дроны больше полагаются на автономность бортовых систем, которые способны автоматически, без участия оператора выполнять заранее запрограммированные стандартные задачи, такие как обход локальных препятствий, аварийный возврат домой и др. Это предъявляет более высокие требования к бортовому компьютеру.

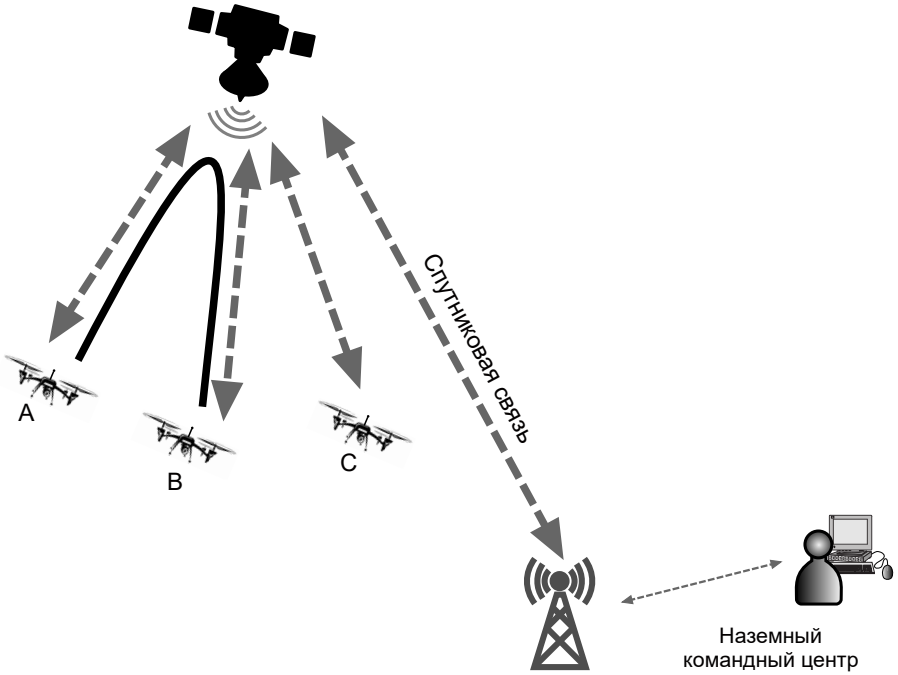


Рис. 27.3. Управление дронами через спутник

Другим способом снятия ограничений на размер зоны действия группы дронов могло бы стать использование нескольких базовых станций, например базовых станций мобильной сотовой сети. Однако это решение не подходит для экстремальных условий, для которых дроны, собственно, и предназначены, когда инфраструктурные объекты, скорее всего, окажутся разрушенными и дронам придется действовать «на свой страх и риск».

Децентрализованная летающая AdHoc-сеть FANET

Рассмотрим сценарий (рис. 27.4), когда дроны, используя какую-либо технологию радиосвязи, получают возможность общаться непосредственно друг с другом, минуя промежуточный узел — базовую станцию или спутник. Так же, как и в предыдущем случае, дроны связаны в сеть, но при этом устраняются все три недостатка, присущие централизованной схеме:

- уменьшаются задержки;
- исключается единая точка отказа и
- потенциально увеличивается зона действия роя дронов.

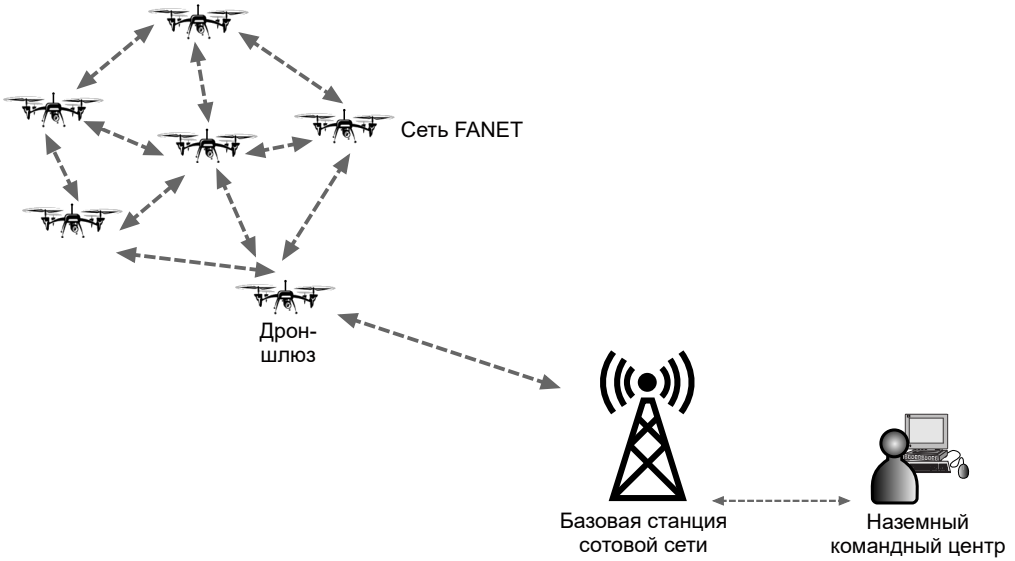


Рис. 27.4. Сеть FANET, состоящая из одной группы дронов

Поскольку теперь для передачи и маршрутизации сообщений между узлами сети инфраструктура (базовая станция) не используется, ее функции по поддержанию работы сети должны быть распределены между всеми дронами — узлами сети. Эти функции реализованы на разных уровнях стека коммуникационных протоколов дрона — от физического (где работает одна из технологий радиосвязи) до прикладного (управляющего миссией роя дронов). Среди прочих, в каждом дроне должны быть реализованы функции формирования сети, определяющие процедуры обнаружения соседей по рою и установления с ними связей — сначала на физическом уровне, а затем и на верхних уровнях, где образуются логические связи, определяющие членство дрона в сети. В результате получаем *децентрализованную* безынфраструктурную сеть, составленную из одноранговых узлов, имеющую ячеистую топологию.

Как мы отмечали в главе 22, большинство сетей без инфраструктуры поддерживают парадигму самоорганизующихся AdHoc-сетей. Для роя дронов, состав которых непостоянен и может быстро изменяться, процедуры формирования сети AdHoc подходят очень хорошо. Узлы сети (дроны) могут быть достаточно удалены друг от друга, так что некоторые из них выходят за границы зон покрытия других узлов. В результате мобильности два узла, находящиеся в некоторый момент времени в пределах зоны досягаемости, через мгновение могут оказаться вне этой зоны. Таким образом, топология сети может изменяться случайным и быстрым образом. В любой произвольный момент времени в зависимости от расположения узлов и паттернов покрытия передатчиков и приемников, уровней мощности передачи и уровней помех существует беспроводное соединение узлов в форме случайного (AdHoc) графа.

Полученная в результате сеть имеет следующие определяющие признаки:

- Она состоит из мобильных, летающих узлов, которые перемещаются в пространстве с достаточно высокой скоростью (до нескольких сотен километров в час).
- Узлы взаимодействуют друг с другом с помощью беспроводной связи.

- Сеть использует принцип AdHoc — свойство, которое позволяет динамичной гибко связанной конструкции перемещающихся узлов самоорганизовываться и спонтанно восстанавливать свою структуру в случае обрыва.
- Сеть поддерживает связи между автономными транспортными средствами, способными летать, и рядом сетевых объектов, развернутых на земле.

Общепринятое в научной литературе название сети, обладающей вышеперечисленными свойствами, — *летающая AdHoc-сеть* (Flying AdHoc Network, **FANET**).

ПРИМЕЧАНИЕ

Отсутствие привязки к базовой станции дает группе дронов, образующей сеть, принципиальную возможность «свободного» полета. Способные обмениваться данными напрямую и координировать свои действия дроны готовы к автономному выполнению миссий, хотя для практического использования этой возможности чаще всего дело не доходит. Одним из ограничений являются энергозатраты: полностью децентрализованная координация дронов для избежания столкновений требует большого объема вычислений, а непрерывные бортовые вычисления быстрее разряжают аккумуляторы. Однако по этой тематике активно ведутся исследования, например, в проекте DARPA OFFSET изучается автономная работа роя, состоящего из более чем 250 дронов.

Связь между роем и инфраструктурой осуществляется через соединение «точка-точка». Для этой цели выделяется дрон, называемый **шлюзовым дроном**, который взаимодействует с землей и передает сообщения другим БПЛА. Шлюзовый дрон оснащен двумя типами трансиверов: один — для связи с другими дронами роя сигналами низкой мощности и на короткие дистанции, другой — для связи с инфраструктурой сигналами высокой мощности и на больших расстояниях. В результате остальные БПЛА в роевом составе нуждаются только в недорогих и легких трансиверах для коротких дистанций. Это не только значительно расширяет диапазон связи сети для обеспечения большого покрытия, но и делает мелкие БПЛА с меньшей полезной нагрузкой более эффективными.

Архитектуры с несколькими группами дронов

Для различных целей могут потребоваться разные типы БПЛА, отличающиеся габаритами, весом, техническими характеристиками и др. Такая неоднородность усложняет работу дронов в одной группе, и для такого случая более подходящей оказывается архитектура с несколькими группами дронов. Здесь мы снова возвращаемся к централизованной схеме, в которой обмен между группами идет через инфраструктуру (на рис. 27.5 дроны *A* и *B*). Однако в пределах группы дроны общаются между собой в режиме AdHoc. Очевидно, что в этой схеме присутствуют недостатки централизованной архитектуры, а именно: сообщения между дронами, принадлежащими к разным группам, идут с задержкой, а инфраструктура остается точкой отказа.

Для смягчения этих недостатков предлагается схема, показанная на рис. 27.6. Здесь несколько групп образуют *иерархическую многоуровневую структуру*. Первый уровень образуют узлы, принадлежащие к одной группе. Они взаимодействуют в режиме AdHoc. На втором уровне происходит связывание узлов, относящихся к разным группам (на рисунке узлы *A* и *B*). Для этого используются узлы-шлюзы, которые передают сообщения между группами, не обращаясь к инфраструктуре и, следовательно, не увеличивая задержку. Отказ инфраструктурного элемента не приводит к полному отказу системы.

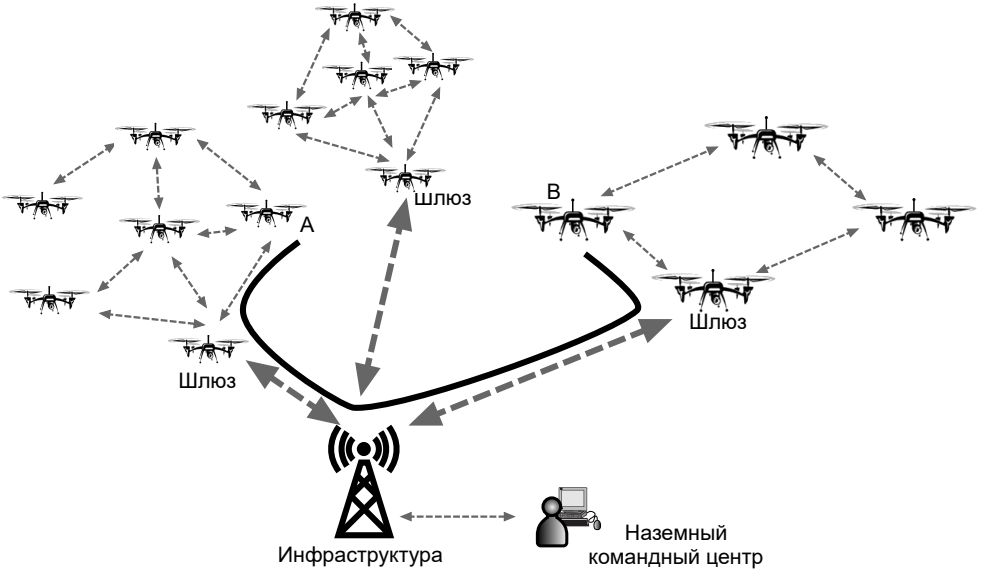


Рис. 27.5. Многогрупповая частично централизованная архитектура

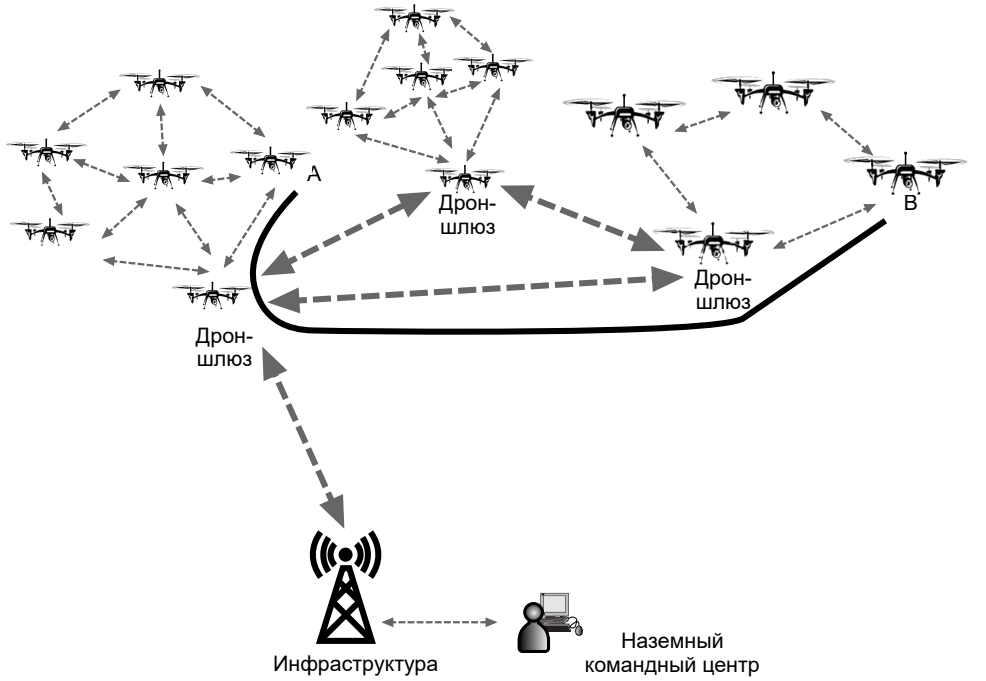


Рис. 27.6. Многогрупповая многоуровневая архитектура

Для систем управления многогрупповым роем дронов характерна *ячеистая топология*.

Такая двухуровневая схема легко масштабируется на большее число уровней с большим числом БПЛА, что повышает надежность и эффективность роя дронов.

Эффективность управления многогрупповыми конфигурациями сетей дронов зависит от того, насколько рационально разделены дроны на группы, часто называемые также кластерами. Каждый кластер состоит из:

- **Головного узла кластера** (Cluster Head, CH): специального БПЛА, который координирует связи внутри кластера и часто служит шлюзом для других кластеров или наземной станции.
- **Членов кластера** (Cluster Member, CM): обычных БПЛА, которые для передачи сообщений членам других кластеров обращаются к своему головному кластеру.

Такая декомпозиция задачи связи дронов на внутрикластерную и межкластерную снижает накладные расходы на маршрутизацию и повышает масштабируемость, позволяя рою достигать десятков или сотен БПЛА. Кроме того, с помощью кластеризации снижаются требования к пропускной способности каналов. Например, в задаче по мониторингу некоторой территории члены кластера передают результаты наблюдения головному дрону, который агрегирует их и передает базовой станции.

Обычно к одному кластеру относят дроны, имеющие близкие характеристики. Например, один кластер могут образовывать дроны, летящие близко друг к другу или обладающие примерно одинаковой мобильностью. Из-за неравномерной мобильности, частого изменения скорости и высоты полета в сетях FANET выполняется частая рекластеризация. Выбор головного узла также может основываться на разных критериях. Например, головным назначается дрон, имеющий самый высокий заряд батареи, или самую стабильную траекторию, или наибольшую связность с остальными дронами кластера. Если центральный дрон при полете выходит за границы зоны покрытия кластера, теряет заряд или по какой-то причине отказывает, в сети оперативно проводятся «перевыборы».

Сети FANET

Общие свойства и различия сетей MANET, VANET и FANET

Сети FANET относятся к семейству беспроводных самоорганизующихся AdHoc-сетей, в которое входят также мобильные AdHoc-сети MANET и транспортные AdHoc-сети VANET, а также некоторые другие[1].

Сети MANET (Mobile AdHoc Network) разработаны для перемещающихся портативных устройств, подобных смартфонам, ноутбукам. лэптопам, так чтобы они могли взаимодействовать друг с другом прямо, не полагаясь на какую-либо инфраструктуру. Термин «MANET» появился в середине 90-х годов при создании рабочей группы IETF, целью которой была модификация существовавших в то время протоколов маршрутизации для

[1] Некоторые авторы рассматривают сети VANET и FANET не как самостоятельную область, исследования, а как подкласс сетей MANET.

мобильных радиосетей. В рамках проекта в числе прочего рассматривалась задача координации групп пехотинцев с помощью сетей портативных радиостанций.

Сети VANET (Vehicular AdHoc Network) предназначены для использования в транспортных системах для обеспечения связи между автомобилями и придорожной инфраструктурой в целях безопасности и удобства водителей, пассажиров и пешеходов.

Как видим, каждый из трех типов мобильных сетей имеет свою область применения, в то же время у всех них много общего:

- Прежде всего, все они являются *AdHoc*-сетями, а это значит, что им присущи свойства сетей этого класса — отсутствие сетевой инфраструктуры, способность к децентрализованным действиям по самоконфигурации, самоподдержке, самоуправлению и быстрому развертыванию.
- Все эти сети характеризуются *ограниченными ресурсами*: сравнительно низкой пропускной способностью беспроводных каналов связи и пониженным уровнем энергообеспечения — их узлы работают на батарейках и аккумуляторах.
- Во всех трех типах сетей узлами являются свободно перемещающиеся мобильные устройства, соединенные беспроводными связями. Узлы в сетях MANET, VANET, FANET являются одноранговыми, каждый из которых совмещает в себе *функции конечного узла и маршрутизатора*. Узлы оснащены беспроводными передатчиками и приемниками с установленными на них антеннами.
- Характерной топологией для этих сетей является *ячеистая многошаговая структура* (AdHoc mesh). Обычно она автоматически формируется после настройки и активации узлов. В то же время некоторые гибридные варианты этих сетей, включающие базовые станции, приводят к конфигурациям типа «звезда», «иерархическая звезда» и «иерархическая ячеистая топология».

По определению в сетях AdHoc (MANET, VANET, FANET) отсутствует предустановленная инфраструктура, такая как базовые станции, маршрутизаторы, точки доступа, а значит, базовые сетевые функции (маршрутизация, передача данных, присоединение к сети) выполняются распределенно самими узлами. Однако на практике узлы мобильных сетей AdHoc часто нуждаются в доступе к информационным сервисам инфраструктурных сетей (услуги интернета, корпоративных систем, облачные вычисления и т. д.) или к управляющим функциям наземных базовых станций. Поэтому сети MANET, VANET и FANET часто имеют **гибридную архитектуру** (рис. 27.7), в которой один или несколько узлов служат шлюзами к внешним инфраструктурным сетям.

На роль шлюза назначается один из одноранговых узлов сети, то есть теперь этот узел помимо функций конечного узла и маршрутизации будет дополнительно выполнять функцию шлюза. В ходе работы сети роль шлюза может переходить к другому узлу. При выборе могут учитываться некоторые особенности узлов, например, это может быть наиболее подходящий узел с топологически наилучшим подключением к инфраструктуре, или узел, снабженный двойным стеком связи. Важно подчеркнуть, что шлюз не является предустановленной инфраструктурой и по-прежнему остается одноранговым узлом.

Объем услуг, предоставляемых AdHoc-сети инфраструктурой, может сильно отличаться в зависимости от организации сети и от типа приложения. В одних случаях управляющая роль инфраструктуры сводится к минимальному набору команд и управляющих сигналов, направляемых наземным центром в адрес роя дронов, который выполняет свой полет практически автономно и нуждается только в экстренном вмешательстве, например в сигналах,

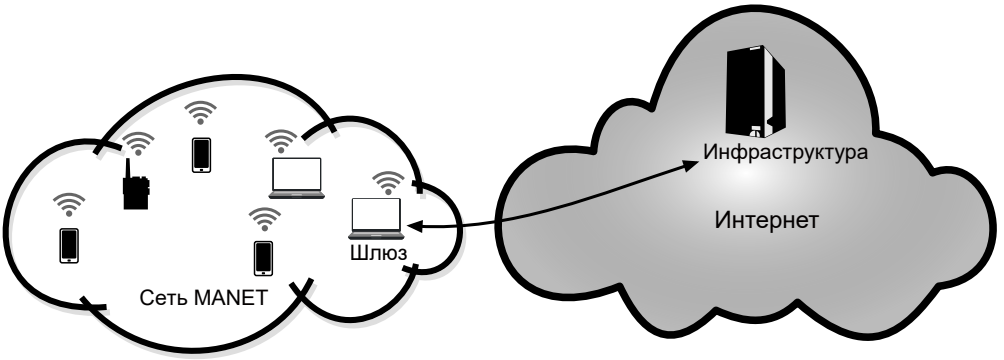


Рис. 27.7. Гибридная сеть MANET

отменяющих миссию. Дроны также используют связь с наземным центром управления очень экономно,только иногда отправляя в центр данные мониторинга.

В других же случаях предусматривается тесное взаимодействие роя с землей: через связь «шлюз — базовая станция» идет интенсивный поток подробных инструкций навигационной системе о параметрах траектории полета, а узлам передаются маршруты, вычисленные наземной системой маршрутизации. В обратном направлении идет диагностическая информация от бортовых датчиков и данные видеонаблюдений.

Несмотря на то что гибридные AdHoc-сети в настоящее время встречаются более часто, сети AdHoc в чистом виде существуют и являются незаменимыми в сценариях, где отсутствует доступная инфраструктура.

Все сети MANET, VANET и FANET имеют мобильные узлы, что приводит к быстроизменяющейся динамической топологии, однако степень ее изменчивости разнится у сетей MANET, VANET и FANET. Именно различия в природе узлов определяют особенности организации соответствующих сетей.

В соответствии с принятыми для MANET соглашениями в качестве узлов могут выступать любые портативные мобильные устройства, которые способен переносить на себе человек, — смартфоны, ноутбуки, лэптопы, военные рации и даже не очень скоростные средства передвижения типа скутера или велосипеда. Примером сети MANET служит беспроводная сеть, составленная из членов спасательно-розыскной команды, оснащенных персональными рациями и передвигающихся пешком по сравнительно небольшой территории. Мобильность узлов в MANET ограничивается снизу скоростью пешехода (5 км/ час), а сверху — скоростью легкого транспортного средства (50 км/час). Подчеркнем, что диапазон 5-50 км/час не жесткое физическое ограничение, а условное соглашение, принятое участниками научных исследований в области MANET для представления уровня мобильности «человеческого масштаба». Но даже при таком ограничении скоростей топология MANET является настолько непредсказуемой и динамичной, что оказалось невозможным использовать для нее существующие традиционные протоколы маршрутизации проводных сетей. Вместо этого были разработаны новые протоколы маршрутизации специально для MANET.

Сети VANET используются для связи автотранспортных средств между собой и с придорожным оборудованием с целью повышения безопасности на дорогах и управления

движением. Транспортные средства создают радиосвязи спонтанно, во время движения по дорогам. Диапазон мобильности для VANET определяется как 10–120 км/час. И хотя скорости перемещения в VANET выше, чем в MANET, топология этой сети меняется более предсказуемым образом за счет того, что узлы-автомобили обычно перемещаются по заранее проложенным и обозначенным на картах дорогам и шоссе, а не по случайным траекториям, как носители портативных устройств сети MANET, бегающие по пересеченной местности. Кроме того, перемещение автомобилей регулируется правилами вождения.

Но и VANET, и MANET по степени мобильности узлов сильно уступают сети FANET. Скорость перемещения дронов в сетях FANET может значительно варьироваться в зависимости от поставленной перед ними задачи. Так, они могут практически не двигаться, зависая, например, над пострадавшим от наводнения районом, куда необходимо доставить интернет, или, напротив, лететь со скоростью, близкой к 500 км/час, при проведении разведки лесных пожаров на большой территории. В последнем случае топология сети FANET становится настолько динамичной, размах взаимных перемещений узлов настолько велик, что возникают частые отключения, обрывы связи и даже разделение сети на несвязанные части. Связи могут образовываться и исчезать не только из-за изменения взаимного положения БПЛА во время полета, но и по причине выхода устройств из строя из-за неисправности или временного отзыва дронов на землю для подзарядки. В отличие от узлов MANET и VANET, которые базируются на земле и перемещаются в двухмерном пространстве, узлы FANET летают в трехмерном пространстве.

Обычно БПЛА расположены друг от друга на расстоянии, большем, чем узлы MANET и VANET, это значит, что пространственная плотность узлов в FANET ниже, чем в MANET и VANET, что усиливает риск разделения сети на несмежные части. В этих обстоятельствах требуются особые схемы маршрутизации, отличные не только от традиционных протоколов проводных сетей, но и от усовершенствованных протоколов сетей MANET. Разработка эффективных и адаптивных алгоритмов маршрутизации, способных выдерживать динамические изменения топологии, является центральной задачей исследователей FANET.

Навигация

Важной особенностью сети дронов является сочетание высокой пространственной подвижности и информационной связности, то есть узлы одновременно участвуют в двух процессах. Во-первых, будучи летающими устройствами, они перемещаются в пространстве, меняя свои координаты в соответствии с определенной для них траекторией полета. Во-вторых, они являются узлами коммуникационной сети, то есть поддерживают между собой информационную связь, передавая друг другу данные через промежуточные узлы в соответствии с заданными адресами получателей по выбранным маршрутам. Первый процесс регулируется навигационной системой, второй – коммуникационными протоколами связи.

В отличие от наземных MANET или VANET, здесь сама подвижность летательных устройств изменяет топологию сети в трехмерном пространстве, поэтому навигация и коммуникация должны координироваться. Представьте себе вашу корпоративную сеть, которая в какой-то прекрасный момент вдруг поднялась в воздух и с огромной скоростью полетела в некотором направлении. Как должен работать протокол маршрутизации в условиях, когда из-за отставания серверов от маршрутизаторов их связи разорвутся?

В каждый момент времени в результате принятых навигационных решений меняется взаимное расположение дронов в трехмерном пространстве, то есть сеть дронов принимает определенную пространственную конфигурацию. Этой конфигурации соответствует определенная топология сети, для которой коммуникационные протоколы должны проложить коммуникационные маршруты и передать вдоль этих маршрутов данные. Задача маршрутизации разрешима, только если топология отвечает определенным требованиям, таким, например, как наличие связности между некоторыми узлами сети. С другой стороны, требования к топологии могут быть выполнены только путем соответствующей навигации, когда для обеспечения связности некоторые дроны должны переместиться в зоны покрытия друг друга. Таким образом, навигационная и коммуникационная системы функционируют взаимосвязанно.

Существуют разные подходы к координации навигационной и коммуникационной систем.

Слабая связь (адаптивность). Навигация работает самостоятельно, рассматривая перемещение дронов как первоочередное условие выполнения миссии. Коммуникационная система подстраивается под текущее пространственное расположение дронов, пытаясь с помощью реактивных протоколов маршрутизации и управления мощностью передатчиков компенсировать ухудшение или потерю связи между парами дронов или частями роя. Средства сетевого уровня на основе навигационной информации могут прогнозировать, куда будут двигаться дроны, и это делает возможной упреждающую корректировку таблиц маршрутизации.

Сильная связь (совместная оптимизация). Планировщик миссии объединяет планирование маршрута и связности сети. Для дронов роя выбираются траектории, которые являются компромиссом между достижением целей миссии роя и сохранением связности коммуникационной сети. Для поддержания оптимальной топологии сети дроны вносят небольшие корректирующие поправки в скорость или пространственные координаты. Алгоритмы планировщика миссии решают задачи многокритериальной оптимизации, например, «максимизировать покрытие области наблюдения, сохраняя при этом связность всего роя дронов или его большей части».

Лидер-ведомый. В этом методе из роя дронов выделяется один дрон — лидер или несколько дронов — ударная группа. Лидер или ударная группа считают своей основной задачей выполнение миссии роя, и поэтому их навигационная система при выборе траектории не принимает во внимание проблему сохранения связности коммуникационной сети. В то же время остальные члены роя считают своей основной задачей сохранение связности коммуникационной сети роя (в том числе с лидером или членами ударной группы) и выбирают свои траектории соответствующим образом.

Функциональная схема дрона — узла FANET

Поскольку БПЛА, являющийся узлом сети FANET, выступает в двоякой роли — летающего аппарата и сетевого устройства, его функциональные блоки должны включать как базовые подсистемы БПЛА, так и сетевые подсистемы, специфичные для FANET (рис. 27.8).

- *Базовые подсистемы БПЛА* (полетная платформа, навигация, автопилот, полезная нагрузка) обеспечивают полет дрона и выполнение порученных ему задач.
- *Специализированные подсистемы FANET* (коммуникационный и сетевой модули, отвечающие за связь, бортовой вычислительный модуль) превращают БПЛА из простого

дрона в мобильный сетевой узел, способный к информационному взаимодействию с другими узлами.

- *Наземное управление* предоставляет пользователю интерфейс для взаимодействия с дроном.



Рис. 27.8. Основные функциональные блоки узла БПЛА в FANET

В число базовых подсистем входят:

- *Полетная платформа* обеспечивает физический полет и удержание БПЛА в воздухе. Она включает несущую конструкцию, которая должна быть легкой и прочной основой для движущей системы — двигателей, регуляторов скорости, пропеллеров, крыльев. Силовая установка представляет собой источник питания в виде аккумулятора, топливного элемента или солнечной батареи. Система управления питанием контролирует потребление энергии в различных подсистемах.
- *Навигационные датчики* (акселерометр, гироскоп, магнитометр, барометр) измеряют ориентацию, угловую скорость, ускорение и высоту полета; GPS/GNSS обеспечивает дрон координатами его местоположения.
- *Датчики восприятия*: камера, лидар[1], радар, ультразвуковые датчики — обеспечивают БПЛА информацией о состоянии окружающей среды, а также обнаруживают препятствия.
- *Автопилот/система управления полетом* обеспечивает устойчивость, безопасность и способность БПЛА автономно выполнять команды. Важным элементом автопи-

[1] Лидар дрона — это система дистанционного зондирования, которая использует лазерные импульсы для сканирования окружающей среды и создания высокоточных 3D-моделей местности.

лота является низкоуровневый контроллер, на вход которого непрерывно поступает информация от навигационных датчиков о текущем положении дрона. На основании этих данных вычисляется отклонение текущего положения от заданных, желаемых параметров полета. По величине отклонения рассчитывается управляющий сигнал для двигателя, который выполняет стабилизацию дрона, например, возвращает наклонившийся дрон в горизонтальное положение. Высокоуровневое управление заключается в планировании движения по точкам маршрута, построении траектории, автоматизации взлета/посадки. Кроме того, эта подсистема выполняет функции по поддержке отказоустойчивости — возврат к исходной точке, зависание на месте, автоматическая посадка при низком заряде батареи.

- *Полезная нагрузка* дрона зависит от конкретной задачи. Это могут быть тепловизоры, фото- или видеокамеры для наблюдения; грузовой отсек, захват, система опрыскивания для сельскохозяйственных работ; дополнительные антенны или релейные модули, если БПЛА используется для расширения зоны покрытия FANET.

Коммуникационный и сетевой модули объединяют в себе функции, которые делают БПЛА сетевым узлом:

- Беспроводные приемники и передатчики, поддерживающие интерфейсы Wi-Fi, LTE/5G, mmWave, UHF/VHF-радиостанции, позволяющие узлам связываться друг с другом.
- Направленные антенны и средства формирования луча, увеличивающие дальность и пропускную способность связей.
- Сетевые протоколы маршрутизации.
- Ретрансляторы данных, выполняющие пересылку трафика между дронами или между дроном и наземными станциями.
- Управление трафиком данных для реализации требований QoS.

Бортовые вычисления наделяют дрон локальным интеллектом, позволяющим ему работать без постоянного наземного контроля. В отличие, например, от радиоуправляемых летающих моделей, дроны обладают гораздо большей автономией: они способны самостоятельно распределенным образом принимать решения в воздухе (например, изменять траекторию), а не получать готовые решения, переданные им с земли централизованно. Для этих целей дрон имеет встроенный процессор/компьютер, в задачи которого входят:

- комплексный анализ показаний датчиков, инерциального измерительного блока, интеграция GPS и машинного зрения для надежной навигации;
- локальное принятие решений, автономный выбор путей;
- предотвращение столкновений, на основе децентрализованных правил безопасности (соблюдение дистанции, корректировка траектории);
- обработка данных полезной нагрузки в реальном времени: сжатие видео, обнаружение объектов, запуск моделей ИИ.

Наземное управление и пользовательский интерфейс предоставляют оператору средства общения с сетью дронов, с помощью которых оператор может планировать миссии, задавать маршруты и зоны поиска, переопределять задания во время полета, собирать и агрегировать данные о полезной нагрузке. Кроме того, наземный пункт управления дает возможность оператору дистанционно проводить мониторинг телеметрии, включая местоположение и состояние как отдельных БПЛА, так и сети в целом.

Стек протоколов FANET

Сеть FANET работает под управлением стека TCP/IP, в который наряду с традиционными для этого стека протоколами IP, TCP, UDP, FTP, SNMP включен ряд протоколов, специально разработанных или адаптированных для FANET (рис. 27.9).

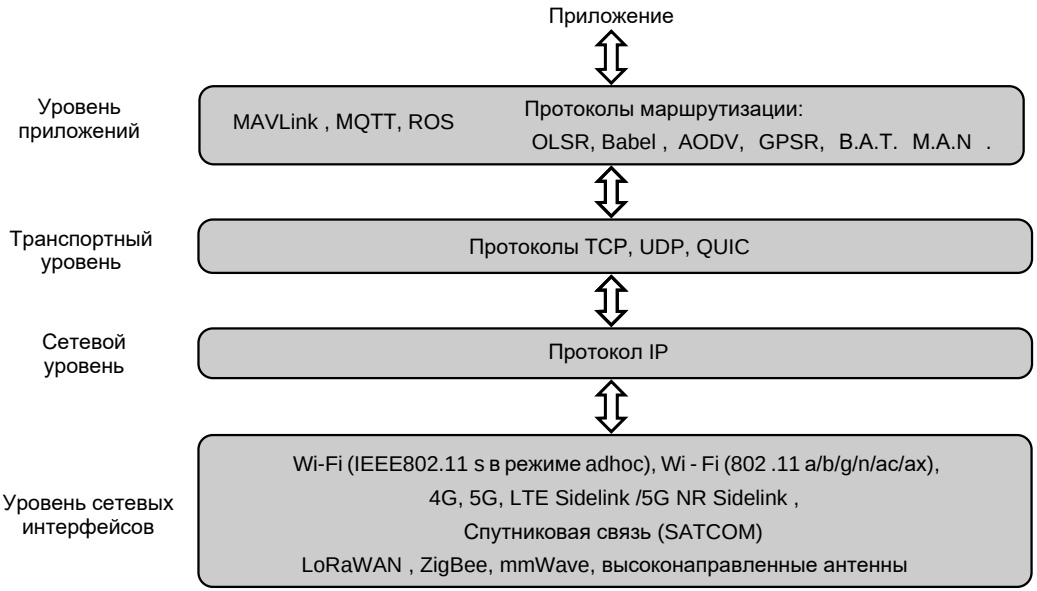


Рис. 27.9. Стек протоколов FANET

Поскольку дроны совмещают в себе функции конечных узлов и маршрутизаторов, на каждом узле сети FANET устанавливается полный стек протоколов.

Уровень сетевых интерфейсов

Как и во всех сетях TCP/IP, технологии, относящиеся к уровню сетевых интерфейсов, решают задачу передачи данных между двумя соседними узлами сети.

В сети FANET существует различие между радиотехнологиями, предпочтительно используемыми для следующих типов связи (рис. 27.10):

- *Связи между дронами* используются, во-первых, для локального обмена данными, а во-вторых, для обмена сигналами координации траекторий полетов и предотвращения столкновений БПЛА в режиме реального времени.
- *Связи между шлюзами* (шлюзовыми дронами) используются для магистрального соединения кластеров БПЛА и передачи агрегированных данных на базовые станции.
- *Связи между дроном и инфраструктурой* напрямую соединяют БПЛА с наземными станциями, вышками сотовой связи или сетевыми контроллерами, то есть со стационарной инфраструктурой. Управление и телеметрия здесь разделены на два параллельных

канала — *нисходящий канал* («инфраструктура — дрон»)[1] управления и команд, для которого требуется сверхвысокая надежность, но допустима низкая пропускная способность, и *восходящий канал* («дрон — инфраструктура») передачи данных полезной нагрузки, для которого, напротив, требуется высокая пропускная способность для видео, датчиков, но не такие высокие требования к надежности.

В то время как связи типа «дрон-дрон» обеспечивают координацию внутри роя, а каналы связи между шлюзами соединяют кластеры, связь «дрон — инфраструктура» привязывает воздушную сеть как единое целое к земле или интернету.

При выборе технологии радиосвязи учитываются особенности этих трех типов связей. Например, для связи «дрон — наземная станция» большое значение имеет скорость передачи данных, так как дрону может потребоваться передавать на землю видеопоток высокого разрешения. Дальность здесь также требуется значительная, чтобы дрон мог удаляться от наземной станции на большое расстояние и в то же время сохранять связь с землей. Для этого типа связи можно выбрать технологию, которая работает только при прямой видимости между дроном и наземной станцией, так как в этом случае между ними маловероятно появление физических препятствий, например, технологию 5G mmWave. А вот для связей «дрон — дрон», которые обычно происходят на близких расстояниях, достаточно эффективной оказывается технология Wi-Fi.

На рис. 27.10 показана модель многоуровневой гибридной сети FANET, в которой дроны уровня роя взаимодействуют через Wi-Fi (IEEE 802.11n/ac/ad в режиме AdHoc), а шлюзовые дроны агрегируют данные роя и пересылают их от шлюза к шлюзу по 5G NR Sidelink (см. далее). Выделенный для коммуникаций с наземным оборудованием дрон подключается к базовой станции, используя миллиметровый канал 5G mmWave.

В табл. 27.1 приведены примеры радиотехнологий, которые наиболее подходят для использования в разных типах связи в сети FANET.

Таблица 27.1. Характеристики связей в сети дронов и требования к технологиям

Тип связи	Особенности связей и требования к технологии	Примеры используемых технологий
Дрон — дрон	• Короткая и средняя дальность (десятки и сотни метров). • Высокодинамичная топология. • Требование низкой задержки. • Работа в режиме ad hoc. • Многошаговые маршруты	• Wi-Fi (IEEE 802.11s в режиме ad hoc) — наиболее распространенный. • Wi-Fi Direct. • LTE Sidelink /5G NR Sidelink. • ZigBee. • LoRaWAN — когда достаточно низкой мощности и низкой пропускной способности
Шлюз — шлюз	• Меньшая мобильность и большая дальность связи по сравнению со связями «дрон — дрон» (шлюзовые дроны часто более стабильны или находятся на большой высоте). • Требуется более высокая пропускная способность и надежность. • Может работать в лицензированном спектре или по спутниковым каналам связи	• LTE Sidelink/5G NR Sidelink — более мощные передатчики, чем «дрон — дрон». • Спутниковая связь (SATCOM) — для связи за пределами прямой видимости. • Wi-Fi (802.11s в режиме ad hoc), но с более мощными радиомодулями и антеннами

[1] В данном случае термины «нисходящий» и «восходящий», определение которых дано в главе 23, могут вводить в заблуждение, потому что они противоречат пространственному расположению дрона и наземной станции.

Тип связи	Особенности связей и требования к технологии	Примеры используемых технологий
Дрон-инфраструктура	• Управление и контроль с наземной станции управления. • Обмен телеметрическими данными (местоположение, статус, параметры полета). • Загрузка данных датчиков или миссии на облачные или периферийные серверы. • Получение обновлений программного обеспечения или перенастройка миссии	• Сотовые сети (4G, 5G,), 5G NR • Wi-Fi (802.11 a/b/g/n/ac/ax), для ближнего радиуса действия. • Спутниковая связь для дальнего радиуса действия. • LoRaWAN — для каналов большой дальности. • mmWave-технологии 5G — для высокоскоростных БПЛА малого радиуса действия

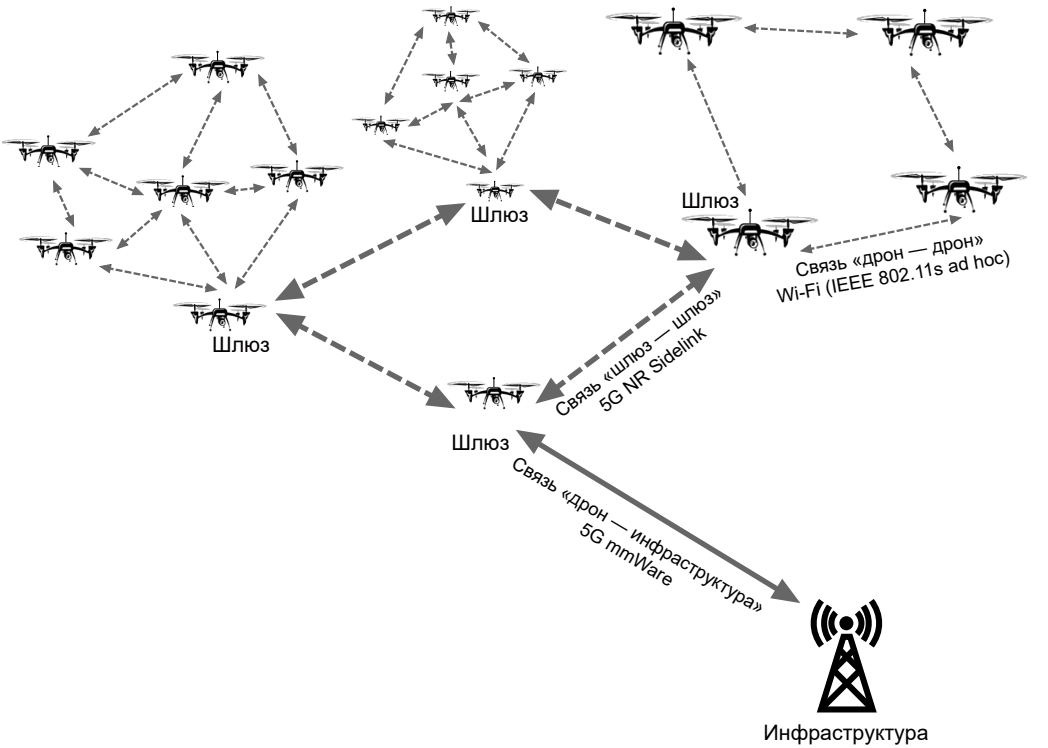


Рис. 27.10. Типы связей и примеры соответствующих технологий в сети FANET

К физическому уровню FANET относится также функционирование антенн, поскольку они влияют на качество связи, покрытие и поддержку мобильности. В отличие от наземных сетей (где часто используются всенаправленные антенны), БПЛА могут использовать на-правленные или формирующие диаграмму направленности антенны, которые фокусируют радиоволны в определенном направлении. Это увеличивает дальность, скорость передачи данных и помехоустойчивость, что особенно полезно в трехмерном воздушном пространтве, где БПЛА быстро движутся и связь может легко оборваться.

Ниже мы коротко рассмотрим особенности использования в FANET технологий, которые уже были изучены в предыдущих разделах этой книги: сотовые сети 4G/LTE и 5G, 5G NR, 5G mmWave, сети Wi-Fi в инфраструктурном и одноранговом AdHoc-режимах, сети Bluetooth и Zigbee, спутниковую связь. Каждая из этих технологий предоставляет средства для непосредственного связывания двух узлов сети FANET, что и является основной функцией нижнего уровня ее стека протоколов.

Сотовые сети в FANET

Сотовые сети **4G**, **5G**, а также их технологии **5G NR** и **5G mmWave** в настоящее время являются наиболее перспективными для каналов «дрон — инфраструктура». Они обеспечивают широкое покрытие и сверхнадежную связь с малой задержкой. Для высотных БПЛА также могут использоваться **5G NTN** (Non-Terrestrial Network) — внеземные сети. 5G NTN, используя дроны как ретрансляторы, расширяют покрытие за пределы наземных базовых станций, доставляя услуги связи 5G в труднодоступные районы, туда, где наземная инфраструктура отсутствует или повреждена.

При использовании сотовой сети для коммуникаций дрона с дроном необходимо, чтобы оба они находились в зоне покрытия этой сотовой сети. Однако такая ситуация имеет место не всегда: дроны могут находиться слишком высоко для сигналов наземных базовых станций сотовых сетей (обычно на высоте более 700 метров сигнал базовой станции становится слишком слабым) или же дроны могут лететь в областях вне зоны покрытия сотовой сети. В таких случаях для связи дронов используются специальные версии технологий связи LTE или 5G NR, предназначенные для взаимодействий D2D (Device To Device). Эти версии называются LTE Sidelink и 5G NR Sidelink.

Режим LTE Sidelink Mode 4 разработан для одноранговой архитектуры сетей LTE, то есть для случая, когда в сети отсутствуют базовые станции и взаимодействие происходит между устройствами пользователя UE (User Equipment) — автомобилями, дронами, лэптопами. В этом режиме используются тот же стек протоколов и те же методы мультиплексирования и модуляции сигналов, что и в основном инфраструктурном режиме, подробно рассмотренном в главе 24. Отличием является только метод доступа к разделяемой радиосреде, который осуществляется не базовой станцией централизованно, а устройствами пользователя по распределенному алгоритму. Устройство пользователя UE *A*, которое хочет послать свои данные устройству UE *B*, начинает прослушивать все ресурсные блоки частотного диапазона, отведенного оператору сети. Если в каком-либо ресурсном блоке не наблюдалось передачи информации на протяжении 1000 мс, то устройство UE *A* считает этот ресурсный блок свободным и оповещает другие станции об этом, передавая в этом ресурсном блоке информацию о его захвате и планируемом расписании его использования, например, каждые 4 тайм-слота. После этого устройство UE *A* начинает передавать свои данные устройству UE *B*, а ситуация коллизии распознается устройством UE *A* по квитанции обратной связи от устройства UE *B*: если квитанция не пришла, то фиксируется коллизия и данные передаются снова (механизм, похожий на тот, который используют сети Wi-Fi). Устройство, захватившее ресурсный блок, может менять расписание его использования или выбрать новый блок, если в данном часто происходят коллизии. Такое расписание доступа называется полупостоянным.

Режим 5G NR Sidelink Mode 2 по принципам работы близок к режиму LTE Sidelink Mode 4, в нем также используется полупостоянное расписание доступа устройств UE к разделяемой среде.

Технологии mmWave миллиметрового диапазона. Диапазоны миллиметровых волн (длина волны 1–10 мм, частота 30–300 ГГц) с недавних пор стали использоваться в сетях 5G, а также в некоторых других технологиях. По сравнению с традиционными вариантами, миллиметровые технологии mmWave имеют следующие преимущества:

- Высокая пропускная способность. В диапазоне свыше 30 ГГц очень много свободных поддиапазонов, поэтому вполне возможно использовать полосы шириной в 100 МГц — 1 ГГц, что при модуляции 1024-QAM дает скорость в 1–10 Гбит/с. Высокая скорость передачи данных идеально подходит для передачи видеопотоков HD/4K или данных с датчиков в беспилотных летательных аппаратах на наземную станцию управления. Направленные лучи снижают помехи и улучшают качество связи.
- Короткая длина волны и компактные антенны. Поскольку размер антенны зависит от длины волны, то приемопередатчики этого диапазоны могут быть оснащены большим количеством антенн, занимающих небольшое пространство, что очень важно для дронов. Например, 100-элементный массив антенн с частотой 60 ГГц может быть интегрирован в 6 см^2.

К недостаткам миллиметровых волн относятся:

- Небольшое расстояние связи. В свободном пространстве потери сигнала миллиметрового диапазона 60 ГГц составляют около 16 дБ/км, поэтому дальность передачи миллиметровых волн с обычной (всенаправленной) антенной составляет всего 20 м. Однако формирование узконаправленного луча с помощью массивов антенн позволяет сконцентрировать мощность в одном направлении и обеспечить дальность передачи до 1 км. Узкая ширина луча означает необходимость точного наведения. Погодные условия — дождь, туман и влажность — могут ослаблять сигналы mmWave.
- Низкая проникающая способность. В то время как сигналы 2,4 ГГц проникают через некоторые твердые материалы, сигналы миллиметрового диапазона легко блокируются большинством объектов. Например, человеческое тело приводит к потерям на миллиметровом диапазоне в 20–50 дБ. При ограничении мощности передатчика (законодательно или из-за ограничений по энергопотреблению на борту дрона) миллиметровый диапазон может не обладать достаточной мощностью для проникновения через препятствия.

Каналы связи mmWave больше подходят для связи «дрон — земля», чем для связей «дрон — дрон». Действительно, между дроном и наземной базовой станцией часто доступна прямая видимость (LoS). Это облегчает критически важную задачу точного наведения луча. Также важно, что в связи «дрон — земля» только одна точка является мобильной. В связи «дрон — дрон» отслеживание луча намного сложнее: обе конечные точки движутся и вращаются в трехмерном пространстве.

Спутниковая связь в FANET

Спутниковая связь (SATCOM), как правило, используется в сетях FANET при выполнении задач за пределами прямой видимости (BLoS). К основным преимуществам, предоставляемым спутниковой связью сетям FANET, можно отнести следующие:

- Дальний радиус действия. Спутники обеспечивают узлам FANET связь на больших расстояниях между удаленными кластерами БПЛА или наземными станциями управления.

- Поддержка высокой мобильности узлов FANET. Спутники обеспечивают непрерывное подключение независимо от движения или высоты БПЛА.
- Повышенная отказоустойчивость сети FANET. В случае сбоя наземной сети спутниковые каналы связи сохраняют непрерывность работы.
- Передача данных в режиме реального времени. Известным недостатком спутниковой связи является слишком большая задержка по сравнению с другими радиотехнологиями. Однако при использовании спутников на низкой околоземной орбите (LEO) задержка может быть уменьшена до 20–40 мс, что позволяет осуществлять телеметрию, потоковую передачу видео и управление в режиме, близком к реальному времени.
- Масштабируемость. Спутниковые системы могут поддерживать большое количество БПЛА.

Рассмотрим, например, гибридную схему взаимодействия сети FANET со спутниками всех трех типов орбит (рис. 27.11):

- Все данные телеметрии, управления и видео с БПЛА в режиме реального времени передаются через низкоорбитальный спутник LEO (например, Starlink).

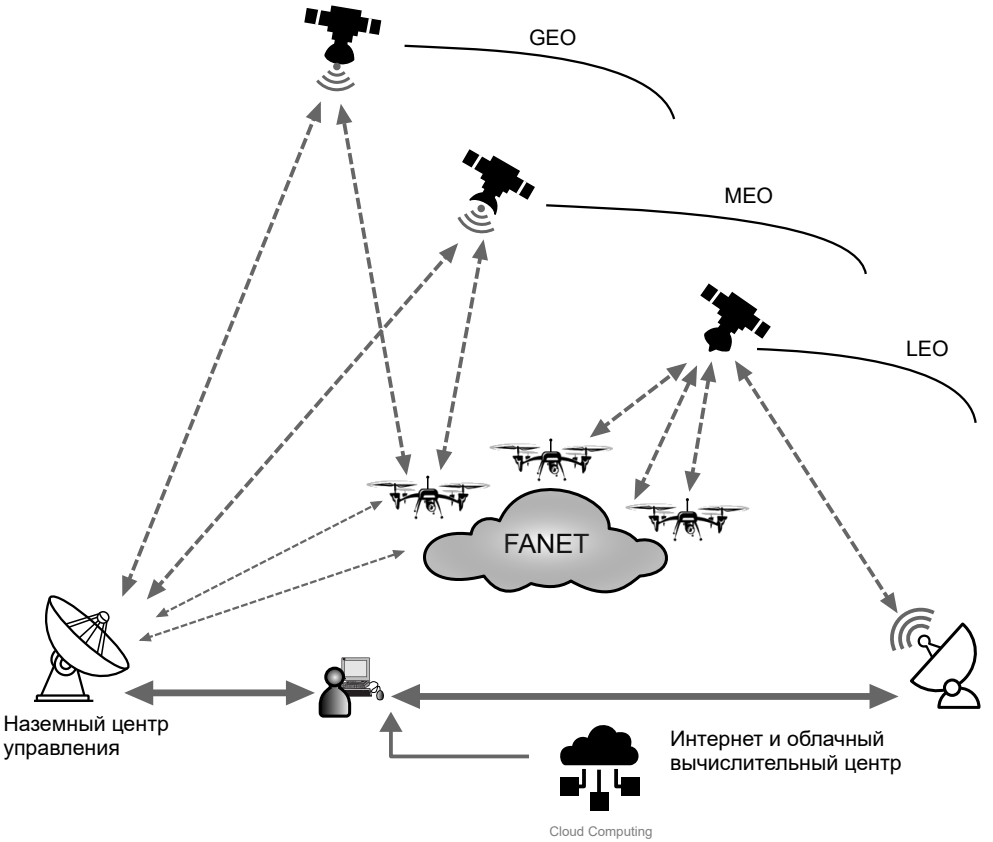


Рис. 27.11. Гибридная сеть FANET со спутниковой связью

- При ослаблении покрытия на низкой околоземной орбите (из-за препятствий или погодных условий) БПЛА автоматически переключаются на среднеорбитальный спутник MEO (например, Iridium), и все данные от дронов теперь идут через него.
- Спутник GEO (например, Inmarsat) служит резервным каналом связи: геостационарная орбита обеспечивает непрерывную связь с БПЛА, хотя и с большей задержкой.

Когда управление дроном осуществляется через спутниковую связь, архитектура дрона должна быть существенно изменена, чтобы соответствовать специфическим задачам и требованиям спутниковой связи. Дрон оснащается терминалом SATCOM, обычно представляющим собой приемопередатчик определенного диапазона в зависимости от конкретного типа спутниковой сети — Inmarsat, Iridium, Starlink. В качестве антенн используются механически управляемая антенна, фазированная решетка или, для систем с низкой пропускной способностью (L-диапазон), всенаправленная антенна.

Некоторые дроны в этой схеме используют несколько типов интерфейсов для поддержания гибридной архитектуры: SATCOM-каналы для работы за пределами прямой видимости и радиоканал для резервный связи на близком расстоянии с наземной станцией управления. БПЛА динамически переключаются между спутниковыми каналами связи (в зависимости от качества сигнала, пропускной способности или приоритета полетного задания), поддерживая одновременно локальную связь друг с другом в сети FANET. Связь между дронами не прерывается даже при обрыве всех спутниковых каналов.

Наземная база управления и облачный вычислительный центр сохраняют и анализируют данные, полученные от гибридной сети, осуществляют планирование и управление миссией.

Такой многоуровневый подход повышает устойчивость, покрытие и пропускную способность.

Wi-Fi в FANET

Технология Wi-Fi наиболее известна как технология построения инфраструктурных сетей с центральным элементом — точкой доступа. Однако она позволяет строить и одноранговые AdHoc-сети, такие как FANET. Для сетей такого типа был разработан специальный вариант технологии Wi-Fi, известной как стандарт 802.11s или Wi-Fi mesh (рассмотренный в главе 25).

Сеть Wi-Fi mesh состоит из узлов разного типа, но для построения сети FANET нужны узлы только одного типа, называемые mesh-узлами. Будем для краткости называть их просто узлами, так как узлы другого типа нас в этом описании не интересуют.

В такой сети узлы совмещают роли конечного узла и маршрутизатора, при этом маршрутизация осуществляется на основе MAC-адресов узлов, то есть на втором уровне. Сетевой уровень для маршрутизации не привлекается.

Узлы поддерживают процедуру образования сети, периодически и широковещательно посылая в радиосреду кадры-маячки, в которые они помещают информацию о себе (MAC-адрес и параметры протокола физического уровня, который они поддерживают, а также идентификатор сети, которую они хотят образовать). В процессе обмена кадрами-маячками между парой узлов устанавливается логическая связь «точка-точка», и из таких пар образуется сеть произвольной ячеистой топологии. Так как не все узлы могут находиться в области покрытия друг друга, то в общем случае топология сети не является полносвяз-

ной, а значит, для взаимодействия некоторой пары узлов может понадобиться маршрут, проходящий через промежуточные узлы, то есть нужна маршрутизация. Протокол маршрутизации для сети Wi-Fi mesh определен, это реактивный протокол маршрутизации R-HWMP; он был описан ранее в разделе «Ячеистая архитектура сети Wi-Fi» главы 25.

На физическом уровне сеть Wi-Fi mesh может использовать любой из стандартов физического уровня семейства 802.11–802.11n, ac, ax, be. Доступ к разделяемой среде также осуществляется стандартным для сетей Wi-Fi способом на основе алгоритма CSMA/CA. Максимальное расстояние между двумя узлами сети при использовании частоты 2,4 ГГц равно около 300 метров для приемопередатчиков стандартной мощности (то есть рассчитанных на применение в пределах дома). При использовании более мощных передатчиков и направленных антенн можно увеличить расстояние до нескольких километров, что хорошо подходит для организации сети дронов.

Еще одним свойством технологии Wi-Fi mesh, полезным для сетей дронов, является то, что она включает в себя проактивный протокол маршрутизации P-HWMP, который создает дерево путей к особому узлу сети Wi-Fi mesh, называемому порталом. Портал предназначен для связи одноранговой AdHoc-сети Wi-Fi с внешней сетью. Эта концепция может быть использована в сети FANET, построенной на основе кластеров, в которой отдельные дроны играют роль шлюзов к наземной станции или другой сети дронов.

В сетях FANET используется еще ряд технологий — Bluetooth, LoRaWAN, ZigBee, Thread и W-SUN FAN, которые были рассмотрены ранее в главах 25 и 26.

Сетевой уровень стека FANET

Протокол IP

В сетях FANET межсетевой протокол IP выполняет те же функции, что и в традиционных IP-сетях, а именно:

- обеспечивает доставку данных между БПЛА через в общем случае неоднородную сеть;
- обеспечивает адресацию — каждый БПЛА получает уникальный IP-адрес;
- выполняет фрагментацию и повторную сборку пакетов;
- взаимодействует с более высокими и низкими уровнями стека протоколов.

Рассмотрим работу протокола IP в сети FANET (рис. 27.12). Пусть, например, дрон A1 отправляет данные (например, видеокадры) дрону-шлюзу Ак:

1. Бортовое программное обеспечение прикладного уровня дрона A1 генерирует данные и передает их нижележащему протоколу транспортного уровня (пусть это будет, к примеру, протокол UDP), а затем ниже по стеку — протоколу IP.
2. Протокол IP инкапсулирует дейтаграмму UDP в пакет, помещая IP-адреса источника (A1) и назначения (Ак) в заголовок.
3. IP-протокол дрона отправителя A1 проверяет, необходима ли маршрутизация. Поскольку шлюз Ак не подключен к A1 напрямую, то маршрутизация нужна. Она выполняется на основе таблицы маршрутизации, которая была сформирована в результате работы одного из протоколов маршрутизации FANET (например, OLSR, AODV, GPSR или BATMAN). В этой таблице протокол IP находит адрес следующего дрона, лежащего на маршруте к шлюзу Ак. Им является дрон A2.

4. Для продвижения данных от А1 к А2 протокол IP обращается к сетевой технологии, работающей на уровне сетевых интерфейсов стека FANET. Пусть это будет технология Wi-Fi.
5. Wi-Fi через беспроводное соединение А1–А2 переправляет IP-пакет дрону А2.
6. Далее процесс повторяется циклически, начиная с п. 3 с той разницей, что вместо дрона А1 отправителем теперь является дрон А2, затем дрон А3, А4 и т. д. На каждом цикле протокол сетевого уровня обращается к технологии нижележащего уровня Wi-Fi для продвижения данных между двумя соседними узлами: А2–А3, А3–А4, ..., А(к–1)–Ак, пока пакет не пройдет весь найденный протоколом маршрутизации маршрут и не достигнет узла назначения — шлюз Ак.

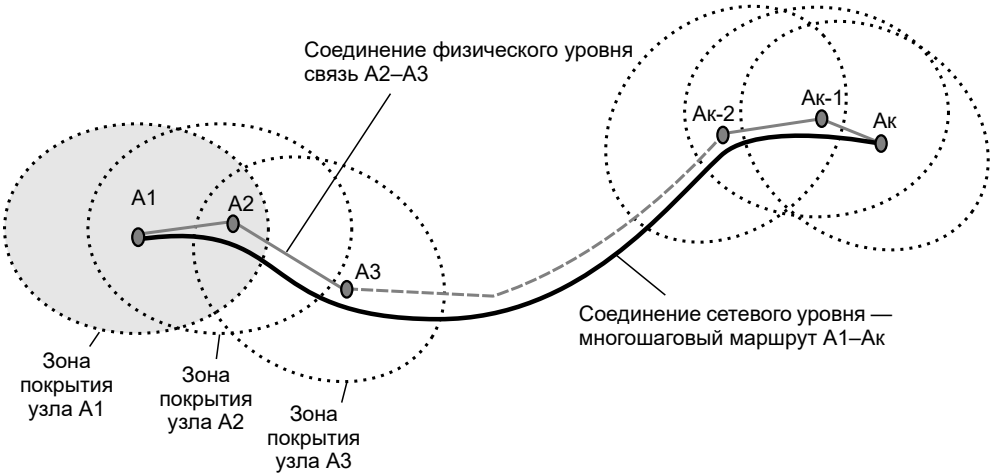


Рис. 27.12. Работа протокола IP по прокладке пути «из конца в конец» на основе соединений «точка-точка"

В соответствии с этим схематичным описанием разница в передаче данных в беспроводных и проводных сетях проявляется только на нижнем уровне стека, где работают радиотехнологии. Однако в реальных условиях особенности сетей FANET — высокая мобильность дронов, частое изменение топологии — вызывают проблемы в работе протокола IP и особенно протоколов маршрутизации, а значит, необходима адаптация существующих протоколов для использования в FANET.

Адресация в FANET

Каждому беспилотному летательному аппарату назначается IP-адрес во время первоначальной настройки сети или в момент присоединения к уже работающей сети. При этом процедура назначения адреса выполняется по-разному для двух сценариев: с подключением к инфраструктуре (например, базовой станции, спутнику или наземной станции управления) и без подключения к инфраструктуре.

В первом сценарии БПЛА получает IP-адрес от инфраструктурного узла, например от базовой станции, на которой работает сервер DHCP. Централизованный характер процедуры гарантирует уникальность адреса. Существует вариант данного инфраструктурного подхода, когда назначением адресов управляет головной или ведущий БПЛА кластера.

Во втором сценарии рой строит сеть автономно, с нуля, используя самоорганизацию по принципу AdHoc. Узлы совместно назначают себе IP-адреса — без центрального DHCP-сервера. Существуют различные распределенные алгоритмы, обеспечивающие уникальность адресов, например схема автоконфигурации интерфейсов IPv6. Здесь дрон генерирует собственный IPv6-адрес, используя префикс сети и идентификатор своего интерфейса (например, MAC). Протокол обнаружения соседей IPv6 ND (Network Discovery) гарантирует, что два БПЛА не выберут один и тот же адрес.

Мобильный IP в сети дронов

Как и в любой другой IP-сети, в сетях дронов существует проблема изменения IP-адреса узла при его перемещении между различными IP-сетями, то есть проблема мобильности. Здесь нужно различать два сценария:

- Дрон связан с наземными пунктами управления *мобильной сотовой сетью*. В этом случае для поддержания мобильности клиентов (здесь — дронов) используются встроенные в сотовую сеть механизмы эстафетной передачи. Однако специфика дронов в некоторой степени ослабляет эффективную работу этих механизмов, ведь они разрабатывались в расчете на перемещение телефонов по горизонтальной плоскости. Дроны же, летая высоко над землей, принимают сигналы одновременно от большого числа базовых станций, что приводит к усилению интерференции сигналов, ненужным переключениям между соседними сотами, быстрому изменению силы сигнала и другим помехам.
- Дрон пользуется услугами беспроводной технологии, в которой *отсутствуют механизмы мобильности*, такой, например, как Wi- Fi, LoRaWAN или ZigBee. В этом случае проблема обеспечения мобильности становится еще более сложной.

Рассмотрим подробнее второй сценарий. Стандартный IP-протокол предполагает, что каждый адрес привязан к узлу сети. Однако когда узел (в данном случае дрон) перемещается между зонами покрытия наземной радиосети, его IP-адрес может измениться. Рассмотрим, например, схему гибридной сети, состоящей из сети FANET и двух наземных связанных между собой через интернет IP-сетей, назовем их для определенности «первая» и «вторая». В каждой из сетей имеется базовая станция, способная поддерживать связь с дронами сети FANET. При создании роя дронов им были назначены IP-адреса от базовой станции первой сети, то есть они являются узлами этой сети. Поставлена задача провести с помощью дронов разведку в зоне катастрофы, а полученные дронами данные должны передаваться через базовые станции в управляющий компьютер, находящийся в интернете. Приложение, решающее поставленную задачу, состоит из серверной части, выполняемой на управляющем компьютере, и клиентских частей — на дронах. Поток данных между клиентской и серверной частями приложения идентифицируется парой сокетов (номер порта, IP-адрес) со стороны отправителя и получателя.

Предположим, что во время полета один из дронов, имеющий адрес IP1, вышел за пределы зоны покрытия первой сети и присоединился ко второй сети, в которой получил другой адрес — IP2 (рис. 27.13). Технически он имеет все возможности продолжать передачу разведывательных данных через вторую базовую станцию. Однако сокет клиентской части приложения, работающей на этом дроне, перестал соответствовать его IP-адресу. В результате нарушилась связь между частями приложения и пакеты (например, квитанции и инструкции) от управляющего компьютера не находят дрон, который улетел из своей домашней сети. Это поставило выполнение полетного задания под угрозу.

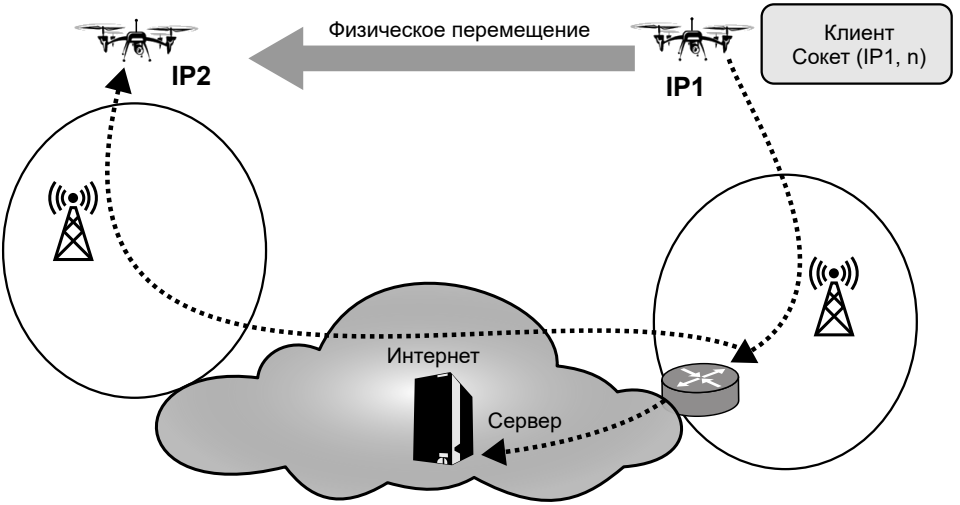


Рис. 27.13. Перенаправление трафика мобильных узлов

Для решения такого рода проблем используется технология Мобильный IP (см. главу 14). Если расположение базовых станций и возможные траектории дронов таковы, что имеется высокая вероятность их перемещения между сетями, то для сохранения мобильности следует в соответствии с технологией Мобильный IP провести установку дополнительных программных модулей этой технологии на дроны, а также на некоторые наземные маршрутизаторы с тем, чтобы трафик приложения прозрачным образом перенаправлялся на переместившийся дрон.

На практике описанный механизм Мобильного IP применяется в сетях дронов только в тех случаях, когда, во-первых, поддержка непрерывных прикладных сессий является *критически важной* и, во-вторых, когда имеется контроль над всей наземной инфраструктурой для установки поддерживающего Мобильный IP программного обеспечения. В остальных случаях от поддержки мобильности просто отказываются, позволяя дрону смену IP-адреса и перезапуская сессии прикладных протоколов при переходе дрона от одной наземной станции к другой.

Прикладной и транспортный уровни

Транспортный уровень обеспечивает сквозную надежность связи и контроль перегрузок. Этот уровень включает известные протоколы UDP (для данных, передаваемых в реальном времени, например телеметрии) и TCP (для надежной передачи файлов или управляющих данных). В последнем специально для применения в мобильных сетях изменен алгоритм контроля перегрузки.

Кроме того, в стеке FANET на транспортном уровне может использоваться протокол QUIC — для работы с высокой мобильностью и прерывистыми соединениями. Этот протокол детально описан в главе 19.

Прикладной уровень в FANET — это уровень «логики миссии». Он определяет, какими данными обмениваются дроны между собой или с наземными станциями управления. Та-

ким образом, прикладной уровень в FANET располагается поверх транспортного (UDP/TCP) и сетевого (IP) протоколов и может использовать стандартные интернет-протоколы или специальные протоколы, разработанные для конкретной миссии.

Уровень приложений обеспечивает специализированные функции и интерфейсы для пользователей или наземных станций управления. Он включает в себя задачи по координации дронов и управление роем, потоковое видео и передачу изображений, сбор данных наблюдения и разведки, управление чрезвычайными ситуациями/мониторинг окружающей среды.

Стек протоколов FANET включает кросс-уровневые средства, которые для выполнения своих задач обращаются к функциям протоколов разных уровней. В частности, к кросс-уровневым относятся задачи планирования пути и предотвращения столкновений, формирование и управление сетью, обеспечение энергоэффективности, мобильности. Мы встречаем примеры кросс-уровневого подхода в маршрутизации, которая учитывает физическое перемещение узлов или же выбирает маршруты, принимая во внимание энергопотребление узлов сети.

Помимо протоколов, специально разработанных для FANET, в этих сетях используются и некоторые стандартные протоколы, например:

- HTTP/HTTPS — протокол взаимодействия веб-сервисов, может использоваться для связи между БПЛА и землей.
- FTP/TFTP — протокол передачи файлов, может использоваться для передачи данных миссии на землю.
- SNMP — протокол управления сетью, применяется для мониторинга состояния или производительности сети БПЛА.
- MQTT (Message Queuing Telemetry Transport) — протокол эффективен для передачи телеметрических и управляющих сообщений между БПЛА и землей.
- CoAP (Constrained Application Protocol) — протокол, аналогичный по функциям протоколу HTTP, но более простой. Хорошо работает в сетях FANET с низким энергопотреблением и низкой пропускной способностью.
- RTP/RTSP — протоколы передачи мультимедийных данных в реальном времени, широко используется для передачи голоса в мобильных сотовых сетях LTE/4G и 5G. В сетях FANET может использоваться для потоковой передачи видео или данных с датчиков между БПЛА или от БПЛА в центры управления.
- NTP/PTP — протоколы синхронизации времени. Обеспечивают синхронизацию часов БПЛА для выполнения группового полета или установки временных меток данных.

Маршрутизация в сетях дронов

Особенности маршрутизации в FANET

Маршрутизация в FANET является ключевой функцией, реализующей поиск наиболее эффективных путей, по которым дроны могут передавать свои сообщения друг другу и инфраструктуре радиодоступа, в частности базовой станции.

Главной особенностью сетей дронов, определяющей то, каким образом решается в этих сетях задача маршрутизации, является высокая мобильность узлов, которая приводит

к интенсивному изменению топологии. Обычные для проводных компьютерных сетей события, приводящие к изменению топологии, — отказ узла, физическое добавление или устранение узла при реконфигурации сети, повреждение кабеля — являются довольно редкими. Напротив, в сетях дронов все перечисленные события случаются гораздо чаще, а кроме того, добавляются новые, вызванные пространственным перемещением узлов. Можно представить, как при передаче сообщения узел, являющийся узлом следующего шага маршрута, вдруг начинает удаляться от узла отправителя с высокой скоростью. Такие перемещения (прилеты-отлеты) могут приводить к ухудшению качества линии связи вплоть до ее разрыва.

В таких условиях работа проактивных протоколов маршрутизации, к которым относятся практически все традиционные протоколы маршрутизации, становится практически невозможной. Частые изменения топологии сети приводят к постоянному интенсивному обмену маршрутными объявлениями, которые затопляют сеть и приводят к неэффективному расходованию ее пропускной способности. Для сетей дронов это означает еще и большие энергетические затраты на постоянные вычисления новых маршрутов.

Гораздо более эффективным решением является использование в беспроводных сетях, и, в частности, сетях дронов, протоколов *реактивной* маршрутизации, когда поиск маршрута к узлу назначения выполняется только по запросу. После установления маршрута он поддерживается до тех пор, пока получатель не станет недоступен или пока маршрут не перестанет использоваться. Ниже мы рассмотрим один из таких протоколов маршрутизации — протокол AODV. Однако реактивные протоколы маршрутизации имеют очевидный недостаток: в то время, когда проактивные протоколы заранее определяют маршруты, после чего они доступны в любой момент времени, реактивные протоколы заставляют пакет ожидать, прежде чем маршрут к узлу назначения будет найден и установлен, и это время ожидания добавляется к задержке передачи данных.

В сетях с мобильными узлами, таких как FANET, информационная связь узлов должна координироваться с их физическим перемещением. Этот факт невозможно игнорировать при выборе маршрутов в сети. Учет перемещения узлов может быть сделан в протоколах маршрутизации путем *прогнозирования* взаимного положения узлов сети в пространстве на основе их пространственных координат, скоростей и направления перемещения. С помощью такой информации можно выбрать лучший промежуточный узел, ближайший к узлу назначения. Для реализации этого подхода система маршрутизации сети дронов должны работать совместно с ее системой навигации.

Протоколы маршрутизации сети дронов должны учитывать энергетические затраты на поиск маршрутов и на передачу данных по маршруту и минимизировать их по возможности.

В сетях дронов может применяться *кластеризация* сети, и протокол маршрутизации должен отражать такую структуру сети. Кластеризация предполагает разделение сети на меньшие зоны или кластеры, где назначенный головным узел кластера контролирует группу дронов, входящих в кластер. В сетях с кластеризацией обычно применяется иерархия протоколов маршрутизации. Внутри кластера применяется протокол маршрутизации, создающий древовидную топологию маршрутов с главным дроном кластера в качестве вершины дерева, и дронами — членами кластера в качестве ветвей и листьев дерева. Между головными дронами применяется другой протокол маршрутизации, который может создавать произвольную ячеистую топологию связей. Мы видели такой двухуровневый иерархический подход в беспроводных сетях Wi-Fi 802.11s, ZigBee и Thread.

В сетях дронов применяется специальный вариант групповой маршрутизации, так называемый геокастинг. **Геокастинг** — это особый случай многоадресной рассылки, которая используется для доставки пакетов группе узлов, расположенных в заданной географической области. С точки зрения реализации геокастинг представляет собой форму «ограниченной» широковещательной рассылки: сообщения доставляются всем узлам, находящимся в заданном регионе. Этого можно добиться, маршрутизируя пакеты от источника к узлу внутри региона геокастинга, а затем применяя широковещательную передачу внутри региона. Протоколы маршрутизации, учитывающие местоположение, используют географические координаты для пересылки данных, полученные, например, от системы GPS. В сетях дронов получение координат от системы GPS является стандартной функцией системы навигации, поэтому такая информация всегда имеется в распоряжении протоколов маршрутизации. Узел выбирает следующий транзитный участок для пересылки пакетов, используя физическое местоположение своих соседей, а также физическое местоположение узла назначения.

Реактивный протокол AODV

Протокол маршрутизации AODV (Adhoc On-Demand Distance Vector) разработан для мобильных сетей AdHoc (MANET). Его разработчики ориентировались на нестабильную топологию таких сетей, которая быстро меняется из-за частого разрыва и появления связей между узлами сети. Для эффективной работы в этих условиях разработчики протокола AODV наделили его свойствами реактивных протоколов, которые строят маршрут между узлами сети только тогда, когда этого требует трафик, а если такой потребности нет, то маршрут и не строится, так что и изменения топологии в такие периоды реактивным протоколом просто игнорируются. Мы уже рассматривали достоинства реактивных протоколов при работе в сетях с быстро меняющейся топологией в главе 16.

Принципы работы протокола AODV

Все узлы сети имеют IP-адреса, но протокол не учитывает объединение узлов в подсети, маршрут находится не к подсети (к префиксу IP-адреса), а к отдельному хосту, имеющему индивидуальный IP-адрес. Говорят, что это маршрутизация, основанная на хостах, а не на сетях. AODV — это дистанционно-векторный протокол с метрикой «количество хопов до узла назначения».

Целью протокола AODV является создание таблицы маршрутизации, имеющей стандартный вид, то есть такой же, как таблица маршрутизации RIP или OSPF, которые мы рассматривали в главе 16. Запись в такой таблице включает такие стандартные поля, как «Адрес назначения», «Следующий хоп», «Количество хопов», «Время жизни записи», а также дополнительное поле «Последовательный номер маршрутного объявления» (Sequence Number, SN), на основе которого создана данная запись таблицы маршрутизации (его назначение мы рассмотрим позже).

В том случае, когда какому-нибудь узлу требуется отправить пакет по некоторому IP-адресу назначения, которого нет в таблице маршрутизации, этот узел посылает широковещательный запрос своим непосредственным соседям (то есть в случае беспроводной сети тем, которые окажутся в момент рассылки в зоне его покрытия), в котором указывает адрес назначения, к которому он ищет маршрут, и свой адрес. Все узлы, получившие этот запрос, просматривают свои таблицы маршрутизации, и, если находят там маршрут к за-

прашиваемому адресу назначения, то отправляют направленный (не широковещательный) ответ запросившему узлу, в котором, как и во всех дистанционно-векторных протоколах, указывается количество хопов до адреса назначения.

В том же случае, если узел не находит запрашиваемого адреса в своей таблице маршрутизации, то он распространяет запрос далее, также широковещательно отправляя его в сеть. Этот процесс повторяется до тех пор, пока какой-то узел не ответит на запрос, или же запрос дойдет до узла назначения, и он пошлет ответ запросившему узлу.

При обработке запроса каждый узел извлекает из этого запроса не только адрес назначения, но и адрес узла, от которого запрос поступил, и создает на его основе запись о реверсивном маршруте к узлу-источнику запроса, пользуясь таким образом информацией, «проходящей мимо». Поэтому ответ источнику запроса всегда передается не широковещательно, а на основе индивидуальных адресов всех промежуточных узлов.

Протокол AODV предназначен для сетей с низким энергопотреблением, поэтому при его создании предпочтение отдавалось простым, не требующим сложных вычислений алгоритмам нахождения маршрутов. Из-за своей простоты эти алгоритмы не всегда находят наилучший маршрут, в то же время они обеспечивают отсутствие маршрутных петель в сети и обладают быстрой сходимостью.

Найденные маршруты имеют сравнительно небольшой срок жизни, поэтому в отсутствии трафика к некоторому узлу маршруты к нему удаляются и таблицы маршрутизации поддерживаются в компактной форме.

В протоколе AODV существуют механизмы, решающие проблемы затопления сети и появления петель маршрутов, и мы рассмотрим их на конкретном примере работы сети, показанной на рис. 27.14. Эта сеть состоит из 8 узлов-дронов, имеющих IP-адреса, условно обозначенные *A*, *B*, *C*, *D*, *E*, *F*, *G* и *H*.

Запрос маршрута

Взаимное расположения узлов в пространстве таково, что области покрытия дронов образуют сеть с топологией, показанной в нижней части рисунка. Будем считать, что в начальный момент таблицы маршрутизации всех узлов пусты, а узлу *A* необходимо отослать IP-пакет узлу *F*. Для того чтобы узнать маршрут к узлу *F*, узел *A* отправляет в сеть запрос — сообщение запроса о маршруте **RREQ** (Route REQuest).

Сообщение имеет широковещательный IP-адрес 255.255.255.255 и следующие поля (в скобках указаны значения, которые узел *A* поместил в соответствующее поле):

- Адрес назначения (F).
- Адрес источника (A).
- B-ID — идентификатор широковещательного запроса (1).
- TTL — поле времени жизни пакета с запросом (2).
- Количество хопов до источника (0).

Поле TTL ограничивает распространение запроса по сети. Каждый узел, приняв запрос, должен вычесть из значения TTL единицу и распространить его (если он не знает ответ) далее только в том случае, если новое значение TTL не стало равным нулю. В нашем примере узел *A* установил TTL равным 2. Обозначим для краткости такой запрос как RREQ (*F*, *A*, 2). Этот запрос является *первой итерацией* поиска маршрута к узлу *F*.

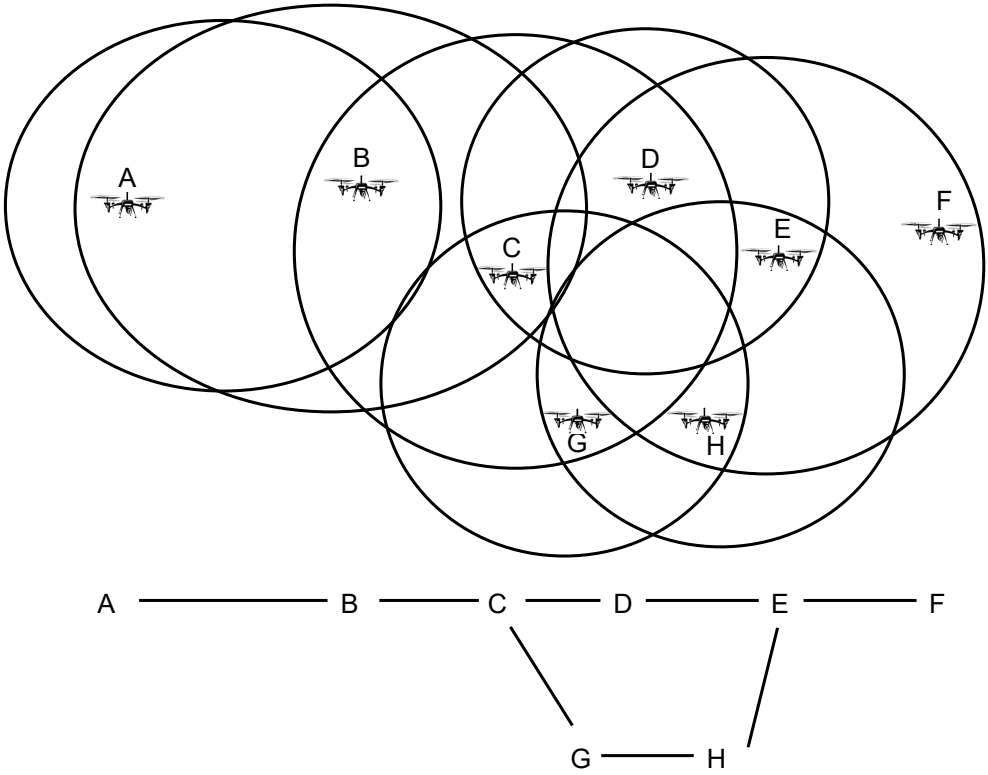


Рис. 27.14. Маршрутизация AODV в сети дронов

Запрос получает только узел *B*, так как все остальные узлы не входят в зону покрытия узла *A*. Узел *B* не находит маршрута к *F* в своей таблице маршрутизации и делает широковещательную ретрансляцию запроса RREQ (*F*, *A*, 1) всем соседям. Он должен это сделать, так как после вычитания единицы TTL стало равным 1 (а не 0). Узел *B* также запоминает **реверсивный маршрут** к узлу *A* по адресу источника из запроса RREQ (*F*, *A*, 2). Он его должен будет использовать для отправки ответа узлу *A*. В записи реверсивного маршрута количество хопов до узла *A* устанавливается равным 1, так как это увеличенное на 1 значение количества хопов до узла *A* в запросе RREQ (*F*, A, 2).

Узел *B* также временно запоминает комбинацию (адрес *F*, адрес *A*, B-ID), чтобы больше не ретранслировать этот запрос, если он его снова получит. Этот механизм временного хранения запросов (кэш запросов) должны использовать все узлы для подавления дубликатов RREQ. Узел *A* также получил дубликат своего запроса от широковещательной рассылки узла *B* и отбросил его, так как сохранил свой запрос в кэше.

Узел *C* получает запрос RREQ (*F*, *A*, 1) от узла *B*. Он не отправляет ответ, так как не знает маршрута к узлу *F* и не делает широковещательной рассылки запроса RREQ (*F*, *A*, 0), потому что после уменьшения TTL на 1 его значение стало равно 0.

В результате запрос проходит путь $A \rightarrow B \rightarrow C$, а узел *A* ответ не получает. На этом заканчивается первая итерация поиска маршрута от *A* к *F*.

Вторая итерация начинается после того, как у узла *A* истек тайм-аут ожидания ответа. Узел *A* снова широковещательно передает пакет RREQ (*F*, *A*, 4) соседям. При этом поле TTL наращено на 2, чтобы расширить область поиска маршрута. Значение идентификатора широковещательной рассылки B-ID наращивается на единицу, чтобы запрос второй итерации можно было отличить от запроса первой итерации и узлы для нового запроса не использовали механизм подавления дубликатов запросов.

Во второй итерации запрос распространяется по двум путям:

$$A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \text{ и } A \rightarrow B \rightarrow C \rightarrow G \rightarrow H.$$

Узлы *B*, *C*, *D* и *G* широковещательно передают запрос RREQ (*F*, *A*, 4-3-2) по цепочке, так как не знают маршрута к *F*, а поле TTL в принятом запросе не нулевое. Одновременно узлы *C*, *D* и *G* заносят в свои таблицы реверсивные маршруты $C \rightarrow A$ (следующий хоп *B*), $D \rightarrow A$ (*C*) и $G \rightarrow A$ (*C*) соответственно. Запрос останавливается на узлах *E* и *G*, так как значение TTL после вычитания единицы становится нулевым. Узел *A* опять не получает ответ.

Третья итерация оказывается для узла *A* успешной: он наращивает поле TTL еще на 2 и теперь его запрос RREQ (*F*, *A*, 6) доходит до узла *F*. Узел *E* в передаче запроса получает его две копии — одну от узла *D*, а другую от узла *H*. Только одна из них передается узлу *F*, та, которая пришла раньше. Запоздавший запрос будет рассматриваться узлом *E* как дубликат и будет отброшен. Пусть узел *E* первым получил запрос от узла *H*, следовательно, именно этот узел он посчитал следующим хопом в реверсивном маршруте к *A*.

Ответ о маршруте

Узел *F*, получив запрос, отправляет ответ о маршруте RREP (Route REPlay). В этом сообщении он указывает два адреса — свой и источника запроса. Сообщение RREP передается по уникальному IP-адресу того узла, от которого был получен запрос RREQ. Это сообщение передается далее по цепочке в обратном направлении от узла назначения узлу источнику, который сделал запрос маршрута.

В ответе RREP узел назначения указывает также *последовательный номер маршрутного объявления SN*. Этот номер играет важную роль в работе протокола AODV. Он предотвращает образование маршрутных петель и ускоряет сходимость. Его устанавливает только источник маршрута, в нашем случае это узел *F*, тогда как другие узлы сети не имеют права изменять этот номер. При создании записи в таблице маршрутизации узел запоминает значение SN из сообщения RREP, на основе которого эта запись создается. Каждый раз, когда какой-либо узел отвечает на новый запрос о маршруте к себе, он увеличивает значение SN, так что остальные узлы могут сравнить новое значение SN с тем, которое они сохранили в своей таблице маршрутизации, и понять, что это самая свежая информация о данном маршруте. Так как значение SN не изменяется в процессе передачи узлу-источнику через промежуточные узлы, то в том, что узлы получают маршрутную информацию «из третьих рук», для них нет опасности поместить устаревшую информацию в свою таблицу маршрутизации, как это происходит, например, в протоколе RIP.

Рассмотрим цепочку передач ответа в нашем примере. Узел *F* генерирует ответ RREP (*F*, *A*, Hop=0, SN=1), выбрав значение SN равным 1, так как ранее он не объявлял маршрут к себе для узла *A*. Узел *F* передает ответ узлу *E*, так как он получил запрос непосредственно от узла *E*.

И здесь возникает интересная ситуация: узел *E* получил две копии запроса — от узла *D* и от узла *H*. Какому узлу передать ответ или, может быть, обоим? Разработчики протокола AODV приняли следующее решение: передавать нужно только одному узлу, тому, от которого запрос был принят первым, а другому узлу (или узлам), чей запрос был признан дубликатом, не передавать. Такое решение упрощает работу протокола AODV, но в жертву принесена эффективность найденных маршрутов, так как не всегда запрос приходит первым от лучшего маршрута. Вот и в нашем примере узел *E* получил первым запрос от узла *H*, поэтому он передает ответ RREP (*F*, *A*, Hop = 1, SN = 1) этому узлу (нарастив число хопов, но не изменив значения SN). В результате цепочка передачи ответа выглядит так: $F \rightarrow E \rightarrow H \rightarrow G \rightarrow C \rightarrow B \rightarrow A$. Соответственно, после прохождения ответа все промежуточные узлы зафиксировали этот маршрут у себя в таблице маршрутизации, так что его полная последовательность от *A* до *F* равна: $A \rightarrow B \rightarrow C \rightarrow G \rightarrow H \rightarrow E \rightarrow F$ и на один хоп длиннее, чем лучший маршрут $A \rightarrow B \rightarrow C \rightarrow D \rightarrow E \rightarrow F$.

После прохождения ответа по всей цепочке от *F* до *A* у узлов этой цепочки появилась запись в таблице маршрутизации об маршруте к адресу назначения *F*, а также записи о реверсивном маршруте к адресу назначения *A*. Например:

Запись в таблице маршрутизации узла *A*:

Адрес назначения	Следующий хоп	Количество хопов	SN
F	B	6	1

Запись в таблице маршрутизации узла C:

Адрес назначения	Следующий хоп	Количество хопов	SN
F	G	4	1
A	B	2	-

Если у некоторого узла в таблице маршрутизации уже имеется маршрут к адресу назначения *X*, то при приходе сообщения RREP с маршрутом к этому же адресу назначения *X* он заменяется на новый (из сообщения RREP):

- если в сообщении RREP значение номера SN больше значения номера SN в таблице маршрутизации (то есть маршрут более свежий);
- если значения номера SN в сообщении RREP такое же, как в таблице маршрутизации, но количество хопов до адреса назначения меньше.

Это правило совместно с правилом назначения значения SN только узлом назначения гарантирует отсутствие маршрутных петель при применении протокола AODV.

В реверсивных маршрутах номер SN в таблицах маршрутизации не хранится, как это показано для реверсивного маршрута к узлу *A* в таблице маршрутизации узла *C*. Реверсивные маршруты используются только для передачи ответа RREP в обратном направлении: у них срок жизни рассчитан на поддержание только этой передачи, после чего они удаляются.

Отказ связи или узла

Реакция протокола AODV на отказ связей или узлов основана на использовании сообщения об ошибке маршрута RERR (Route ERRor). В нем указывается адрес назначения,

маршрут к которому больше не доступен. Это сообщение посылается в том случае, когда узел обнаружил потерю связи с одним из своих непосредственных соседей. Протокол AODV узла может обнаружить факт отказа двумя способами:

- Получить сообщение об отказе от протокола нижнего уровня, например Wi-Fi или LTE/5G.
- Использовать периодические сообщения HELLO (они опциональны).

Пусть в нашем примере связь между узлами *H* и *E* отказала. Узел *H* обнаруживает этот факт и видит, что этот отказ приводит к невозможности использовать маршрут к узлу *F* (узел *E* — это следующий хоп этого маршрута). Тогда узел *H* формирует сообщение об отказе маршрута RERR (*F*) и отправляет его узлу *G*, потому что он запомнил, что передавал этому узлу сообщение RREP (*F*, *A*), так что узел *G* пользовался этим маршрутом и ему надо дать знать, что маршрут более не работоспособен. Узел *G* помечает маршрут к *F* как недействительный и передает сообщение RERR (*F*) узлу *C*, которому он в свое время передал сообщение RREP (*F*, *A*). В результате сообщение RERR(*F*) проходит в обратном направлении по маршруту $A \rightarrow F$, и этот маршрут становится недействительным во всех его промежуточных узлах.

Использование явного сообщения об отказе маршрута обеспечивает быстрое реагирование сети с протоколом AODV на обрывы связей или отказы узлов.

Существуют версии протокола AODV, наделенные свойствами *географического протокола маршрутизации*. Например, в недавнем стандарте RFC 9854 (октябрь 2025 г.) в этот протокол добавлены: управление направленным лучом антенны для передачи широковещательных сообщений только определенным узлам (дронам) в определенном секторе луча, учет позиции, скорости и направления движения узла при выборе следующего шага маршрута, например, если соседний узел находится близко, но он удаляется с большой скоростью от узла-источника, то такой узел исключается при передаче запроса на маршрут, вводится новая метрика — стабильность линии связи. При вычислении метрики стабильности учитывается сила сигнала, скорость и направление движения для того, чтобы предсказать, как долго еще эта связь будет существовать.

Маршрутизация в сетях, устойчивых к разрывам (DTN)

Узлы сети FANET обладают сверхмобильностью. Размах их перемещения в трехмерном пространстве достигает десятков метров, скорость полета может быть весьма высокой. При этом их взаимные перемещения могут плохо координироваться.

Как уже было сказано, высокая мобильность узлов сети вызывает высокую изменчивость топологии сети. Она изменяется как по причине пространственного перемещения узлов, так и из-за изменения состояний связи между узлами: меняются характеристики, такие как расстояние, пропускная способность. Из-за превышения пределов дальности беспроводной радиосвязи, нехватки энергетических ресурсов, ограничений полосы частот или зашумленности эфира связи могут рваться, иногда на короткое время, а иногда надолго.

Изменение топологии, в свою очередь, влияет на функционирование сети. Некоторая степень изменений может быть купирована обычными протоколами. Например, при разрыве связи протокол маршрутизации находит другой маршрут, а возникшая при этом за-

держка или потеря пакета компенсируется механизмами транспортного протокола TCP. Однако если в результате разрывов связи сеть деформируется настолько, что отдельные ее части становятся несвязными, то неминуемо возникают задержки, которые превышают пороговые значения транспортного протокола, пакеты, адресаты которых не могут быть достигнуты, обычно отбрасываются, и приложения терпят крах.

Такие фатальные изменения топологии сети возникают во время экстремальных условий работы сети: во время природных, техногенных или военных катастроф. В таких условиях перестают работать и другие средства связи. И тогда возникает вопрос: можно ли каким-то образом использовать эту деформированную сеть дронов? Может ли она принести пользу даже тогда, когда значительная часть ее связей разорваны, а большая часть узлов потеряна, так что сеть стала разряженной, что, в свою очередь, может ухудшить состояние некоторых связей?

Очевидно, что ответ отрицательный, если речь идет о приложениях, работающих в реальном времени. Однако есть другой класс задач, в которых допускается ожидание ответа в течение длительного периода, например почта. Для такого рода приложений при отсутствии альтернативных возможностей в условиях катастрофы приемлемыми оказываются задержки, измеряемые часами.

Однако чтобы сделать сети ограниченной связности пригодными для использования, необходимо снабдить их специальными средствами, ориентированными на преодоление последствий разрыва связей. Например, в такой сети не могут работать традиционные протоколы маршрутизации, в которых маршруты находятся как непрерывная последовательность узлов из конца в конец. В сетях с разрывами связей путь к узлу назначения может оканчиваться на нескольких первых шагах, и как дальше продвигать сообщение — не ясно. А значит, нужно находить другие нетрадиционные методы доставки сообщений адресату.

Сети, предназначенные для работы в условиях, когда связи подвергаются разрывам и из-за этого невозможно обеспечить непрерывное сквозное соединение, называются **сетями, устойчивыми к разрывам** (Delay-Tolerant Networking, **DTN**). Концепция сетей DTN была определена в начале 2000-х в исследовательских работах, посвященных перспективам создания межпланетных сетей, но к настоящему времени она оказалась очень продуктивной в более приземленной области — в разработке гетерогенных сетей с повышенной живучестью, предназначенных для работы в условиях катастроф.

Одним из наиболее эффективных методов связи в условиях, когда обычная связь отсутствует, считается устойчивая к разрывам **сеть дронов**, продолжающая функционировать при потере связности, пусть и с пониженной эффективностью. Приложения такой сети должны выдерживать задержки, превышающие задержки, возникающие при обычной IP-пересылке. То есть в сетях DTN разрывы должны обрабатываться сетью как штатная ситуация, для которой предусмотрено специально разработанное программное обеспечение, в частности, протоколы маршрутизации. Однако для сетей, значительную часть времени пребывающих в виде нескольких «лоскутов», не только традиционные, но даже специально ориентированные на использовании в FANET протоколы маршрутизации оказываются непригодными. К решению этой задачи привлекаются различные, в некоторых случаях революционные подходы. Рассмотрим несколько из них:

1. Когда соединение трудно или невозможно установить за приемлемое количество времени, данные, готовые для передачи, временно запоминаются на исходном или каком-либо промежуточном узле. Они могут перемещаться и храниться по всей сети, пока не

возникнет возможность передать их адресату. Мы уже встречались с таким подходом **«сохранение с продвижением»** (store-and-forward), когда изучали буферизацию пакета в коммутаторе. Однако, в отличие от коммутатора, в котором пакеты буферизируется на очень короткое время, в сетях, устойчивых к разрывам, время хранения должно быть значительно дольше. Пакеты должны храниться в буфере до тех пор, пока узел не сможет скопировать их на один или несколько соседних узлов. Поэтому узел должен иметь достаточную буферную емкость для хранения данных. Если сообщение не может быть доставлено немедленно из-за разделения сети, то для передачи сообщения выбираются те узлы, которые имеют наибольшую вероятность успешной доставки сообщения, по крайней мере те из них, которые могут переместить пакет ближе к месту назначения. Если узлы ничего не знают о состоянии сети, то лучшее, что они могут сделать, — это случайным образом пересылать пакеты своим соседям.

2. Одним из простейших подходов к увеличению вероятности успешной передачи сообщения является **лавинная**, **или эпидемическая, маршрутизация**, в основе которой лежит создание **нескольких копий** передаваемых данных. Лавинная маршрутизация используется в сетях, в которых значительная часть узлов находятся в неактивном состоянии. При необходимости передать сообщение узлы создают несколько копий сообщений и пересылают их доступным в настоящий момент активным узлам. Эти узлы работают как *ретрансляторы* сообщений. Ретрансляторы передают сообщения другим активным узлам-ретрансляторам, как только получают такую возможность, не анализируя, насколько высока вероятность успешной доставки через этот узел. Пакеты, полученные узлом-ретранслятором, пересылаются всем или некоторым соседям этого узла, за исключением того, откуда они пришли. Таким образом, сообщения быстро распространяются по подключенным частям сети, и при достаточном количестве случайных обменов все узлы в конечном итоге получают все сообщения за минимальное количество времени. При больших буферах и низкой нагрузке сети эпидемическая маршрутизация очень эффективна и обеспечивает минимальную задержку и высокую вероятность успеха, поскольку сообщения достигают пункта назначения несколькими путями. Основная проблема эпидемической маршрутизации заключается в том, что сообщения распространяются по всей сети, чтобы достичь только одного пункта назначения, а значит, ресурсы сети используются нерационально.
3. Усовершенствованием лавинной маршрутизации, в которой данные передаются первому попавшемуся узлу, является подход, в котором промежуточные узлы прежде, чем передать сообщение, **оценивают вероятность** того, что отправленное сообщение в конечном итоге достигнет пункта назначения. Основываясь на этой оценке, промежуточные узлы решают, передавать ли данные дальше или продолжать сохранять пакет и ждать более благоприятной ситуации. Примером такого протокола маршрутизации для сетей с устойчивостью к разрывам является протокол PROPHET, который делает оценку основываясь на реалистичной предпосылке, что мобильность узлов не является чисто случайной. Создатели протокола предполагают, *что узлы в DTN, как правило, посещают одни места чаще, чем другие, и что пары узлов, имевшие повторные контакты в прошлом, с большей вероятностью будут иметь контакты в будущем.*
4. Очень интересным подходом (однако применимым только к сетям с мобильными узлами) является **маршрутизация с возможностью управления движением узлов**. Вместо пассивного ожидания переподключения к какому-то более или менее подходящему узлу в этом подходе предлагается *активный поиск контакта.* Рассмотрим, например, сеть,

в которой имеются особые **узлы-паромы**. Эти узлы специально выделены для активного поиска контактов и имеют большую емкость памяти и источника энергии. Возможны два варианта. В первом случае узел-паром переносит и собирает данные, перемещаясь по заранее запланированному маршруту. Другие узлы при необходимости приближаются к парому и взаимодействуют с ним, загружая или выгружая данные. Во втором варианте узел-паром сам объезжает все узлы, которые обратились к нему с запросами. Перемещаясь от одного узла к другому, он выполняет требуемые операции по передаче данных. Управляемая переправа может использоваться в сетях БПЛА при аварийном восстановлении, где БПЛА могут быть оснащены коммуникационными устройствами, способными хранить большое количество сообщений, и им можно приказать следовать траектории, которая соединяет разъединенные части сети.

Вопросы к части VI

1. Назовите главные особенности беспроводных сетей.
2. Какие из следующих утверждений верны:
   а) сеть AdHoc всегда является беспроводной;
   б) сеть AdHoc иногда включает предустановленную инфраструктуру;
   в) принцип AdHoc состоит в способности сети самоорганизовываться;
   г) топология сети AdHoc всегда полносвязная;
   д) сеть AdHoc никогда не включает предустановленную инфраструктуру;
   е) все сети, в которых отсутствует инфраструктура, являются сетями AdHoc?
3. Что влияет на размеры и форму зоны покрытия сети:
   а) взаимное пространственное расположение всех узлов сети;
   б) мощность передатчиков и чувствительность приемников;
   в) рельеф местности;
   г) тип и направленность антенн;
   д) логическая топология;
   е) физическая топология?
4. Какие из следующих утверждений верны:
   а) область покрытия всегда является связной областью;
   б) сеть с полносвязной топологией является одношаговой;
   в) область покрытия узла имеет форму шестиугольника;
   г) сеть с полносвязной топологией, у которой устранены некоторые связи, является многошаговой?
5. Какие приемы могут быть использованы для экономии энергозатрат в беспроводных сетях:
   а) использование батарей с повышенным зарядом;
   б) минимизация периодов активности узлов;
   в) разработка коммуникационных протоколов с учетом энергозатрат;
   г) оптимизация мощности передатчика;
   д) снижение помех для уменьшения количества повторных передач;
   е) увеличение скорости передачи данных?
6. Частоты какого диапазона выше: видимого света или микроволнового?
7. Вследствие какого эффекта электромагнитные волны следуют поверхности земли?
8. За счет чего технологии расширенного спектра уменьшают влияние помех на полезный сигнал?

9. Как определяется коэффициент усиления антенны?
10. Какой размер должен быть у антенны-диполя?
11. Вставьте пропущенные слова: «Пространственное мультиплексирование увеличивает ..., а пространственное разнесение увеличивает ...» (скорость, надежность).
12. Какой вариант FHSS эффективнее уменьшает влияние помех — быстрого или медленного расширения частоты?
13. За счет чего возможно выделить сигнал отдельного канала из общего сигнала в мультиплексировании CDMA?
14. Какой должна быть частота модуляции сигнала в подканалах OFDM, если известно, что эти подканалы сдвинуты относительно друг друга на 20 кГц?
15. В чем состоит негативный эффект засвеченной станции?
16. Названия метода доступа в сетях Wi-Fi и Ethernet отличаются одной буквой: CSMA/CD и CSMA/CA. Поясните, что за этим стоит.
17. Каким образом синхронизируются станции в распределенном режиме доступа?
18. За счет чего достигается увеличение дальности связи устройств в Bluetooth Long Distance?
19. Предложите вариант организации сот с повторным использованием четырех частот. Сравните этот вариант с использованием трех частот. В каком случае сигналы различных сот интерферируют меньше?
20. Как можно увеличить количество одновременно обслуживаемых абонентов в некоторой области покрытия мобильной сети?
21. Почему сообщения пейджинга в сети LTE передаются одновременно в нескольких сотах?
22. Как можно узнать номер своего мобильного телефона:
    а) позвонить со своего телефона на другой мобильный телефон;
    б) используя дополнительное оборудование, просмотреть содержимое SIM-карты;
    в) найти номер телефона в «Настройках»;
    г) получить доступ к домашней базе данных провайдера HSS?
23. Сколько ресурсных блоков сети LTE нужно выделить логическому каналу, чтобы его скорость передачи данных была равна 2,016 Мбит/с при методе кодирования 64-QAM?
24. С какой целью в сети LTE предусмотрено использование двух GTP-туннелей: «базовая станция — шлюз SGW» и «шлюз SGW — шлюз PGW»?
25. Какие технологии используются для виртуализации сети 5G?
26. В чем преимущества и недостатки нового диапазона частот mmWave в сетях 5G?
27. Какие из перечисленных ниже новшеств появились в мобильных сетях LTE:
    а) передача голоса в цифровой форме;
    б) услуга доступа к интернету;
    в) полный переход на протоколы стека TCP/IP;
    г) введение SIM-карт;

д) мультиплексирование OFDMA;

е) использование частотного диапазона mmWave?

28. Как происходит адресация ресурсных блоков данных, которые базовая станция посылает телефону?
29. Какие операции выполняются в процессе присоединения телефона к сети LTE:

а) пользователь вводит имя сети и пароль;

б) телефон входит в посимвольную и покадровую синхронизацию с базовой станцией;

в) аутентификация пользователя выполняется на основе уникального идентификатора абонента IMCI;

г) телефон сообщает сети свой IP-адрес;

д) создаются индивидуальные туннели GTP для данного пользователя?

30. Какие функции выполняет система IMS:

а) аутентификация абонента мультимедийных услуг;

б) установление сессии между телефонами вызывающего и вызываемого абонентов;

в) передача оцифрованного голоса;

г) передача IP-адреса телефона вызываемого абонента сети LTE?

31. За счет какого механизма процедура эстафетной передачи в сети 5G более эффективна, чем в сети LTE?
32. Могут ли станции непосредственно передавать друг другу кадры в сети BSS Wi-Fi?
33. В каком сценарии используются все четыре адреса в кадре Wi-Fi:

а) при передаче кадров между станциями двух BSS, соединенных беспроводной связью;

б) при передаче кадров между станциями двух BSS, соединенных проводной связью;

в) при передаче кадров между станциями одной и той же BSS?

34. Почему при способе агрегирования кадров **A-MSDU** разрешается агрегировать только кадры, адресованные одной и той же станции?
35. За счет чего в сети Wi-Fi достигается приоритетное обслуживание голосового трафика:

а) у этого класса трафика самый короткий межкадровый интервал;

б) у этого класса трафика увеличено время возможной передачи данных;

в) кадрам этого класса дается более высокий приоритет в очереди?

36. В чем состоит основное отличие перехода станции Wi-Fi от одной точки доступа к другой (роуминг) от процедуры эстафетной передачи телефона в сетях LTE/5G?
37. Какие топологии поддерживаются сетями Wi-Fi:

а) звезда;

б) иерархическая звезда;

в) ячеистая;

г) полносвязная?

38. Какие технологии базируются на спецификации IEEE 802.15.4:

а) ZigBee;

б) Thread;

в) Wi-SUN FAN;

г) LoRaWAN?

39. В спецификации IEEE 802.15.4 определены следующие типы узлов:

а) полнофункциональные;

б) с ограниченной функциональностью;

в) PAN-координатор;

г) координатор;

д) точка доступа;

е) конечный узел.

40. Какие характеристики приложимы к технологии LoRaWAN:

а) беспроводная технология;

б) технология с очень низким энергопотреблением;

в) поддерживает связи с центральным узлом на расстояниях 10–15 км;

г) обеспечивает связь на сравнительно низких скоростях (1 Кбит/с – 20 Кбит/с);

д) неэффективно использует разделяемую среду.

41. В чем состоит главное достоинство чирп-модуляции технологии LoRaWAN?

42. В чем состоят сходство и отличие понятий «сеть дронов» и «интернет вещей»?

43. Какими из перечисленных ниже свойств обладает сеть FANET:

а) сеть имеет децентрализованную структуру;

б) узлы сети могут перемещаться со скоростью 500 км/час;

в) в качестве инфраструктурного элемента используется базовая станция;

г) сеть способна к спонтанному восстановлению;

д) сеть поддерживает как проводные, так и беспроводные связи;

е) узлы сети могут практически стоять на месте.

44. Какими свойствами обладают сети MANET, VANET, FANET? Заполните таблицу,

	Ad Hoc	Примеры узлов			Степень мобильности		
		Дрон	Автомобиль	Рация	Низкая	Средняя	Высокая
MANET							
VANET							
FANET							

45. Как координируют свою работу навигационная и коммуникационная системы дрона?

46. Какие из указанных в таблице радиотехнологий могут быть использованы для разных типов связей в сети FANET? Заполните таблицу.

	Wi-Fi (802.11s в режиме Ad Hoc)	LoRaWAN	Wi-Fi (802.11 a/b/g/n/ac/ax)	4G/5G	Спутниковая связь	Диапазон mmWave технологии 5G
Дрон – дрон						
Шлюз – шлюз						
Дрон – инфраструктура						

47. Каким образом дрон может получить IP-адрес:
    а) дрон получает IP-адрес от базовой станции;
    б) назначением адресов управляет головной дрон кластера;
    в) узлы совместно назначают себе IP-адреса;
    г) дрон генерирует собственный IPc6-адрес;
    д) назначением адресов занимается модуль Mobile IP?

48. Какие приемы используются для маршрутизации пакетов в сетях FANET:
    а) проактивный подход при построении таблицы маршрутизации;
    б) прогнозирование взаимного положения узлов;
    в) минимизация энергетических затрат;
    г) иерархия протоколов маршрутизации;
    д) геокастинг;
    е) взаимодействие с системой навигации.

49. Как предотвращается образование маршрутных нетель при использовании протокола AODV?

50. Какими свойствами обладают сети, устойчивые к разрывам (DTN):
    а) они хорошо подходят для решения задач реального времени:
    б) это сети с очень прочными связями;
    в) в этих сетях часто невозможно установить сквозное соединение;
    г) они способны функционировать при потере связности;
    д) в протоколе маршрутизации для сетей DTN может учитываться факт, что узлы, имевшие повторные контакты в прошлом, с большей вероятностью будут иметь контакты в будущем;
    е) для передачи сообщений в сетях DTN могут использоваться узлы-паромы?

Часть VII

Безопасность компьютерных сетей

- Глава 28. Основные понятия и принципы информационной безопасности
- Глава 29. Технологии аутентификации, авторизации и управления доступом
- Глава 30. Технологии безопасности на основе анализа трафика
- Глава 31. Атаки на транспортную инфраструктуру сети
- Глава 32. Безопасность программного кода и сетевых служб

Эта часть книги открывается главой 28, в которой рассматриваются *фундаментальные* понятия и технологии, лежащие в основе любой системы информационной защиты: конфиденциальность, доступность, целостность, уязвимость, угроза, атака, ущерб, аутентификация, авторизация, контроль доступа. Эти понятия поясняются примерами некоторых распространенных пассивных и активных атак на транспортную систему и программное обеспечение сети: отказ в обслуживании, ответвление трафика, спуфинг, шпионы, кража личности. В качестве важнейшей концепции представлен системный подход к обеспечению безопасности — привлечение средств самой различной природы: законодательства, административных мер, процедур управления персоналом, средств физической защиты и программно-технического оборудования. Такой подход включает также следование универсальным принципам построения системы защиты: непрерывности и разумной достаточности, эшелонированного характера и сбалансированности. Завершает главу обзор криптографических методов, являющихся краеугольным камнем практически всех методов информационной безопасности.

Глава 29 посвящена *конкретным* технологиям аутентификации, авторизации и контроля доступа. Рассматриваются процедуры, основанные на одноразовых и многоразовых паролях, цифровых сертификатах и цифровой подписи, описываются мандатный, дискреционный и ролевой способы управления доступом. Приводятся примеры как локальных, так и централизованных систем аутентификации и авторизации.

В главе 30 показано, как различные способы фильтрации могут быть использованы для экранирования сегментов сети от вредительского трафика. Рассматриваются системы мониторинга трафика на основе анализаторов протоколов и агентов протокола NetFlow, которые позволяют распознать атаку путем выявления отклонений образцов трафика от стандартного поведения. Изучаются различные типы файерволов: без запоминания состояния, с запоминанием состояния, с трансляцией адресов (NAT), с функцией прокси-сервера. Рассматриваются особенности анализа трафика и событий системами обнаружения вторжения (IDS). Дается типовая схема разбиения крупной корпоративной сети на зоны безопасности.

В главе 31 изучаются уязвимости и методы защиты транспортной инфраструктуры сети. Особое внимание уделяется безопасности службы DNS и протоколу маршрутизации BGP — двум очень важным элементам архитектуры Интернета, обеспечивающим связность составляющих сетей и узлов в глобальном масштабе. В главе также подробно рассматривается технология защищенного канала IPSec.

В заключительной главе 32 изучаются типичные уязвимости программного обеспечения компьютерной сети, а также приводятся рекомендации по их устранению . Освещены различные методы обнаружения и обезвреживания вредоносного кода: троянских программ, червей, вирусов и программных закладок . Значительная часть этой главы посвящена вопросам безопасности сетевых служб — вебслужбы, почты, облачных вычислений .

ГЛАВА 28 Основные понятия и принципы информационной безопасности

Идентификация, аутентификация и авторизация

Для пояснения таких базовых понятий информационной безопасности, как «идентификация», «аутентификация» и «авторизация», представим информационную систему (далее также — ИС) в виде упрощенной модели контролируемого доступа, когда несколько пользователей совместно работают с ресурсами информационной системы. Родившаяся полвека назад и направленная на повышение эффективности применения компьютера концепция разделения ресурсов выдвинула проблемы безопасности вычислительных систем на первый план — необходимо было *контролировать доступ* пользователей к компьютеру, защищая системные и пользовательские данные от ошибочных или злонамеренных действий. Контролируемый доступ является важным направлением обеспечения безопасности наряду с другими средствами безопасности: криптографической защитой, аудитом, сегментацией сети и др.

В модели контролируемого доступа определены объекты, субъекты, операции и система контроля доступа (рис. 28.1).

Объекты представляют физические и логические информационные ресурсы ИС. К физическим ресурсам относятся как отдельные устройства целиком (процессор, внешние устройства, маршрутизаторы, коммутаторы, физические каналы связи и др.), так и физические разделяемые ресурсы устройств (разделы и секторы диска, процессорное время, физические соединения канала связи). Логическими ресурсами являются файлы, вычислительные процессы, сетевые сервисы, приложения, пропускная способность каналов связи и т. п.

Субъекты представляют сущности, между которыми разделяются информационные ресурсы. Это могут быть легальные пользователи ИС: персонал, поддерживающий работу ИС, внешние и внутренние клиенты; группы легальных пользователей, объединенных по различным признакам. Пользователь осуществляет доступ к объектам ИС не непосредственно, а с помощью прикладных процессов, которые запускаются от его имени. Поэтому в качестве субъектов выступают также прикладные вычислительные процессы. Иногда оказывается полезным представление в качестве субъектов и системных вычислительных процессов.

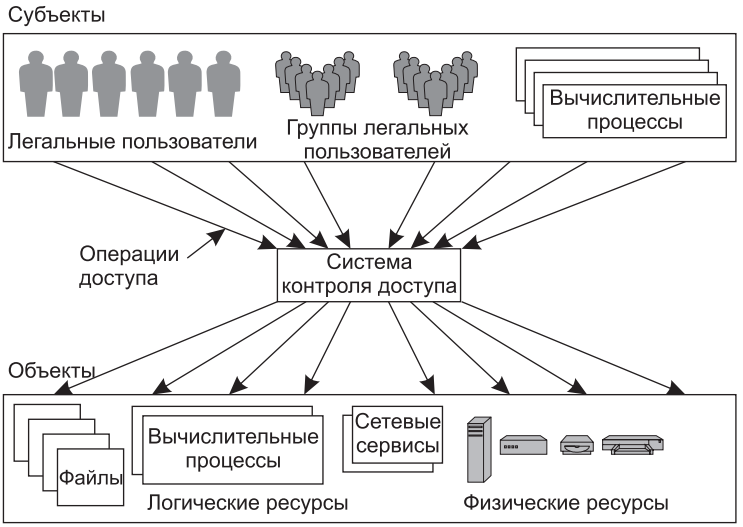


Рис. 28.1. Модель контролируемого доступа

Операции выполняются субъектами над объектами. Для каждого типа объектов существует собственный набор операций, которые с ними может выполнять субъект. Например, для файлов это операции чтения, записи, удаления, выполнения; для принтера — печать, перезапуск, очистка очереди документов, приостановка печати документа; для маршрутизатора — конфигурирование и т. д.

Система контроля доступа решает, какие операции разрешены для данного субъекта по отношению к данному объекту. Для автоматизированного контроля доступа необходимо, чтобы для каждой пары «субъект — объект» были однозначно определены *правила доступа*, на основании которых система могла бы принимать решение о разрешении или запрете выполнения каждой из предусмотренных для данного объекта операций.

Важнейшими элементами управляемого доступа являются процедуры идентификации, аутентификации и авторизации.

Идентификация — это присвоение объектам и субъектам ИС уникальных имен — **идентификаторов**.

Только при наличии уникальных идентификаторов система получает возможность распознавать и оперировать субъектами и объектами. Одни идентификаторы автоматически генерируются операционной системой (ОС) и приложениями (идентификаторы процессов, идентификаторы логических сетевых соединений), другие назначаются администратором компьютерной сети (идентификаторы пользователей, адреса компьютеров, доменные имена сетевых сервисов), третьи порождаются обычными сетевыми пользователями, обладающими таким правом (выбор собственного имени, назначение имен файлам).

Идентификация пользователей представляет собой процедуру, выполняемую при логическом входе в систему, когда пользователь в ответ на выведенное на экране приглашение

печатает свой идентификатор (имя), а система, сверяясь со своими данными, определяет, входит ли данное имя в число имен зарегистрированных (легальных) пользователей.

Пользователь может быть представлен в системе в виде нескольких субъектов и, соответственно, иметь несколько пользовательских идентификаторов. Например, один идентификатор он может применять во время сетевой регистрации, а другой — для работы с корпоративной базой данных.

Аутентификация[1] — это процедура доказательства субъектом/объектом того, что он есть то, за что (кого) себя выдает.

Аутентификация, или, другими словами, процедура установления подлинности, может применяться как к пользователям, так и другим объектам и субъектам, в частности, к данным, программам, приложениям, устройствам, документам (рис. 28.2).



Рис. 28.2. «В интернете никто не узнает, что ты собака, если успешно пройдешь аутентификацию» (рисунок Питера Штайнера)

Аутентификация данных означает доказательство их подлинности, то есть того, что они поступили в неизмененном виде и именно от того человека, который объявил об этом.

В процедуре аутентификации участвуют две стороны:

- *аутентифицируемый* доказывает свою аутентичность, предъявляя некоторое доказательство — *аутентификатор*;
- *аутентифицирующий* проверяет эти доказательства и принимает решение.

Аутентификация бывает односторонней и двусторонней (взаимной).

Так, мы имеем дело с *односторонней аутентификацией*, в частности, при выполнении логического входа в защищенную систему. После того как пользователь сообщает системе свой идентификатор, он должен пройти процедуру аутентификации, то есть доказать, что именно ему принадлежит введенный им идентификатор (имя пользователя). Аутентифи-

[1] Термин «аутентификация» (authentication) происходит от латинского слова authenticus, которое означает «подлинный», «достоверный», «соответствующий самому себе».

кация предотвращает доступ к сети нежелательных лиц и разрешает вход для легальных пользователей.

В качестве аутентификатора аутентифицируемый может продемонстрировать знание некоего общего для обеих сторон секрета, например слова (пароля), или обладание неким уникальным предметом (физического ключа) либо предъявить собственные биохарактеристики (допустим, отпечатки пальцев).

В некоторых случаях односторонней аутентификации оказывается недостаточно, и тогда используют *двустороннюю аутентификацию*. Например, пользователь, обращающийся с запросом к корпоративному веб-серверу, должен доказать ему свою легальность, но он также должен убедиться сам, что ведет диалог действительно с веб-сервером своего предприятия. Другими словами, сервер и клиент должны пройти процедуру взаимной аутентификации. Здесь мы имеем дело с двусторонней *аутентификацией на уровне приложений*. При установлении сеанса связи между двумя устройствами также часто предусматриваются процедуры взаимной *аутентификации устройств* на более низком, канальном, уровне.

Авторизация[1] — это процедура контроля доступа субъектов (пользователей, вычислительных процессов, устройств) к объектам (например, файлам, приложениям, сервисам, устройствам) и предоставления каждому из них именно тех прав, которые для них определены правилами доступа.

В отличие от аутентификации, которая позволяет распознать легальных и нелегальных пользователей, авторизация касается только *легальных* пользователей, успешно прошедших процедуру аутентификации.

Доступ к объектам, полученный в обход разрешений системы контроля доступа, называется *несанкционированным* или *неавторизованным*.

Модели информационной безопасности

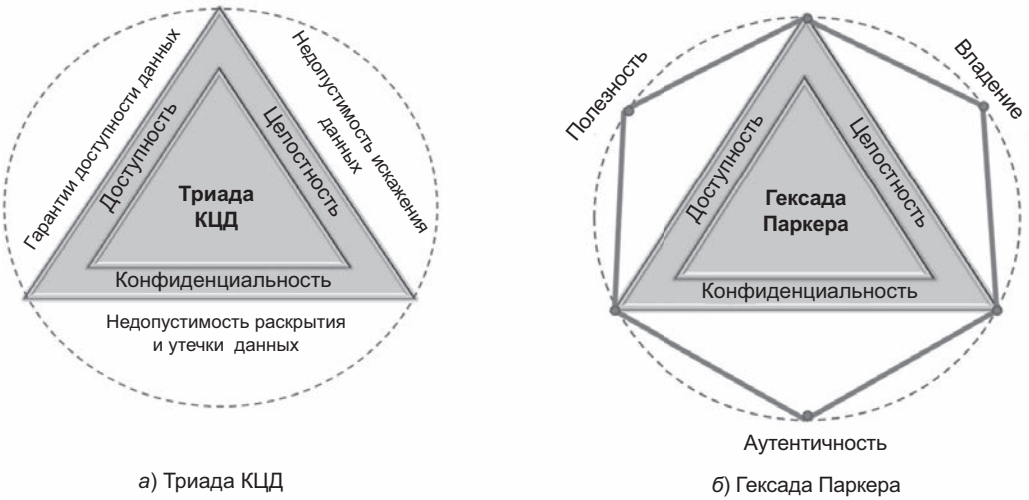
Понятие информационной безопасности может быть пояснено с помощью так называемых **моделей безопасности**, суть которых в следующем: множество всех видов нарушений безопасности делится на несколько базовых групп таким образом, чтобы любое возможное нарушение обязательно можно было отнести по крайней мере к одной из этих групп. Затем система объявляется безопасной, если она способна противостоять каждой из этих групп нарушений.

Триада «конфиденциальность, доступность, целостность»

Одной из первых и наиболее популярных по сей день моделей безопасности является модель, предложенная Зальцером (Saltzer) и Шредером (Schroeder). Авторы постулировали, что все возможные нарушения информационной безопасности всегда могут быть отнесены

[1] Термин «авторизация» (authorization) происходит от латинского слова auctoritas, показывающего уровень престижа человека в Древнем Риме и соответствующие этому уровню привилегии.

по меньшей мере к одной из трех групп: нарушения конфиденциальности, нарушения целостности или нарушения доступности (рис. 28.3, *а*).



а) Триада КЦД

б) Гексада Паркера

Рис. 28.3. Модели безопасности

Соответственно, ИС находится в *состоянии безопасности*, если она защищена от нарушений конфиденциальности, целостности и доступности, где:

- **конфиденциальность** (confidentiality) — состояние ИС, при котором информационные ресурсы доступны только тем пользователям, которым этот доступ разрешен;
- **целостность** (integrity) — состояние системы, при котором информация, хранящаяся и обрабатываемая этой ИС, а также процедуры обработки информации не могут быть изменены, удалены или дополнены неавторизованным образом;
- **доступность** (availability) — состояние системы, при котором услуги, оказываемые системой, могут гарантированно и с приемлемой задержкой быть предоставлены пользователям, имеющим на это право.

Для ссылки на триаду иногда используют аббревиатуру КЦД (**к**онфиденциальность, **ц**елостность, **д**оступность), в англоязычной форме — CIA.

Требования к безопасности могут меняться в зависимости от назначения ИС, характера используемых данных и типа возможных угроз. Трудно представить систему, для которой нарушения целостности и доступности не представляли бы опасности. Вместе с тем обеспечение конфиденциальности не всегда является обязательным.

Например, если вы публикуете информацию в интернете на веб-сервере и вашей целью является сделать ее доступной для самого широкого круга людей, то конфиденциальность не требуется. Однако требования целостности и доступности остаются актуальными. Действительно, если вы не предпримете специальных мер по обеспечению целостности системы, то злоумышленник может изменить данные на вашем сервере и нанести этим ущерб вашему предприятию. Преступник может, например, внести изменения в помещенный на веб-сервере прайс-лист, что негативно отразится на конкурентоспособности вашего пред-

приятия, или испортить коды свободно распространяемого вашей фирмой программного продукта, что, безусловно, скажется на ее деловой репутации. Если бы модифицированные данные были к тому же секретными, то в таком случае имело бы место не только нарушение их целостности, но и их конфиденциальности.

Не менее важным в данном примере является и обеспечение доступности данных. Затратив немалые средства на создание и поддержание сервера в интернете, предприятие вправе рассчитывать на отдачу: увеличение числа клиентов, количества продаж и т. д. Однако существует вероятность того, что злоумышленник предпримет атаку, в результате которой помещенные на сервер данные станут недоступными для тех, кому они предназначались. Примером таких злонамеренных действий может служить «бомбардировка» сервера пакетами, каждый из которых в соответствии с логикой работы соответствующего протокола вызывает тайм-аут сервера, что в конечном счете делает его недоступным для всех остальных запросов.

Некоторые виды нарушений безопасности могут быть приведены к модели КДЦ только путем расширенного толкования основополагающих понятий конфиденциальности, доступности и целостности. В частности, свойство конфиденциальности по отношению, например, к устройству печати можно интерпретировать так, что доступ к устройству имеют те и только те пользователи, которым этот доступ административно разрешен, причем они могут выполнять только те операции с устройством, которые для них определены. Свойство доступности устройства означает его готовность к работе всякий раз, когда в этом возникает необходимость. В свою очередь, свойство целостности может быть интерпретировано как свойство неизменности параметров данного устройства.

Гексада Паркера

За почти полвека, прошедших с момента публикации статьи Зальцера и Шредера, информационные системы и среда, в которой они функционируют, претерпели революционные изменения, поэтому неудивительно, что появились новые типы нарушений, которые намного труднее (если вообще возможно) трактовать в терминах КДЦ. Рассмотрим, например, ситуацию, когда легальный клиент банка посылает по электронной почте запрос на снятие со счета крупной суммы, а затем заявляет, что этот запрос, который хотя и был послан от его имени, он не отправлял. Является ли это нарушением безопасности? Да. Были ли при этом нарушены конфиденциальность, доступность или целостность? Нет. Следовательно, список свойств безопасной системы следует расширить, добавив к КЦД еще одно свойство — неотказуемость. **Неотказуемость** (non-repudiation) — это такое состояние системы, при котором обеспечивается невозможность отрицания пользователем, выполнившим какие-либо действия, факта их выполнения, в частности, отрицания отправителем информации факта ее отправления и/или отрицания получателем информации факта ее получения.

Одной из наиболее популярных альтернатив триаде КЦД является так называемая **гексада Паркера** (Parkerian Hexad), в которой определено шесть базовых видов нарушений, в число которых, помимо нарушений конфиденциальности, доступности и целостности, входят еще три вида нарушений: аутентичности, владения и полезности (рис. 28.3, *б*).

Аутентичность (authenticity) — это состояние системы, при котором пользователь не может выдать себя за другого, а документ всегда имеет достоверную информацию о его

источнике (авторе). Из этого определения видно, что аутентичность является аналогом *неотказуемости*.

Владение (possession) — это состояние системы, при котором физический контроль над устройством или другой средой хранения информации предоставляется только тем, кто имеет на это право.

Полезность (utility) — это такое состояние ИС, при котором обеспечивается удобство практического использования как собственно информации, так и связанных с ее обработкой и поддержкой процедур. В безопасной системе меры, предпринимаемые для защиты системы, не должны неприемлемо усложнять работу сотрудников, иначе последние будут воспринимать меры защиты как помеху и пытаться при всякой возможности их обойти.

Российский государственный стандарт[1] дает определение информационной безопасности на основе гексады Паркера:

> **Информационная безопасность** — все аспекты, связанные с определением, достижением и поддержанием конфиденциальности, целостности, доступности, неотказуемости, подотчетности, аутентичности и достоверности информации или средств ее обработки.

Уязвимость, угроза, атака

Уязвимость (vulnerability) — это слабое звено ИС, которое, став известным злоумышленнику, может позволить ему нарушить ее безопасность. Уязвимостями являются, например, ошибка в программе, примитивный пароль, неправильное назначение прав доступа к файлу с важными данными и множество других дефектов в разработке, эксплуатации или настройке системы.

Уязвимости системы могут быть скрытыми, то есть еще не обнаруженными, известными, но только теоретически, или же общеизвестными и активно используемыми злоумышленниками. Для общеизвестных уязвимостей в программных продуктах производители регулярно выпускают исправления, называемые **патчами** (от англ. patch — заплатка). Так, компания Microsoft даже назначила специальный день — каждый второй вторник каждого месяца, когда она объявляет о новых исправлениях в семействе ОС Windows. Многие из этих исправлений направлены на устранение уязвимостей. Однако к этой рутинной процедуре — регулярному внесению исправлений — не все и не всегда относятся с должным вниманием, из-за чего общеизвестные, но неисправленные ошибки в программном обеспечении являются одним из самых распространенных типов уязвимостей.

Другим типом уязвимостей, которым часто пользуются злоумышленники, являются ошибки в конфигурировании программных и аппаратных средств. Например, имена «администратор» и «гость», установленные по умолчанию во многих ОС, могут облегчить злоумышленникам доступ к системе, поэтому уже при начальном конфигурировании ОС они должны быть заменены другими, менее очевидными именами. С этой же целью администратор должен настроить подсистему интерактивного входа на то, чтобы она не показывала последнего имени пользователя, систему аудита (то есть настроить ее

[1] ГОСТ 13335-1:2006 «Информационная технология. Методы и средства обеспечения безопасности. Часть 1. Концепция и модели менеджмента безопасности информационных и телекоммуникационных технологий».

таким образом, чтобы подсистема фиксировала все успешные и неуспешные попытки входа пользователей), а также выполнить другие, столь же простые, но необходимые настройки.

Поиск уязвимостей — важная часть задачи обеспечения безопасности. Эта работа включает в себя регулярное тестирование системы. В любой момент времени для любой системы можно указать множество различных видов уязвимостей. Например, для ОС и приложений новые уязвимости появляются чуть ли не каждый день. Выявлять их вручную — задача очень трудоемкая, поэтому для автоматизации поиска уязвимостей используют различные программные инструменты — **средства сканирования уязвимостей**, такие, например, как McAfee, Nessus и др. Сканирование заключается в последовательном (адрес за адресом узла, или номер за номером порта, или идентификатор за идентификатором сетевого соединения) направлении запросов целевой системе. Затем на основании полученных ответов генерируется «информационный отпечаток», и, наконец, сравнением «отпечатка» с записями в базе данных выполняется идентификация уязвимости.

Другими базовыми понятиями информационной безопасности являются угроза и атака.

Угроза (threat) — набор обстоятельств и действий, которые *потенциально* могут привести к нарушению безопасности системы (то есть к нарушению ее конфиденциальности, целостности и доступности, если пользоваться моделью КЦД).

Атака (attack) — это реализованная угроза.

Мы в основном ограничимся рассмотрением только *технических* угроз, то есть угроз, исходящих из искусственно созданного человеком мира техники и технологий (в частности, из интернета), не принимая во внимание угрозы, возникающие от природных катаклизмов, военных действий, террористических атак или экономических потрясений.

Атака может произойти только тогда, когда одновременно существуют уязвимость и направленная на использование этой уязвимости угроза (рис. 28.4).

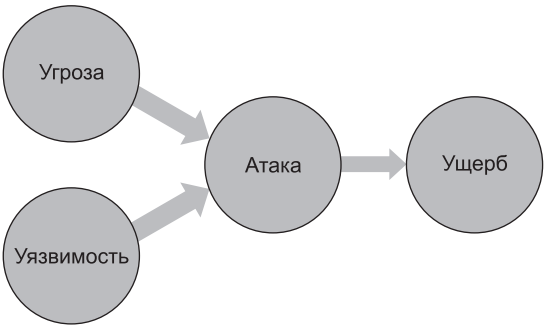


Рис. 26.4. Логическая связь между понятиями «уязвимость», «угроза», «атака», «ущерб»

То есть вполне возможна ситуация, когда система имеет некую уязвимость, но эта уязвимость еще не стала известной злоумышленникам. В подобном случае соответствующая угроза отсутствует, а значит, и атака не может быть проведена. Аналогично существование общеизвестной угрозы не влечет никакой опасности для системы, в которой нет соответствующей уязвимости. Например, появление информации о некоторой ошибке в коде

ОС Windows может породить угрозу, но атака не осуществится, если эта уязвимость будет быстро устранена.

Таким образом, любая угроза направлена на поиск и/или использование уязвимостей системы. В некоторых случаях злоумышленник работает «на ощупь», пытаясь обнаружить тот или иной дефект системы. Система реагирует на такого рода угрозы выдачей сообщений о мелких, но странных неполадках, а также флуктуациями в статистических характеристиках работы системы, на основании которых администратор сети или специалист по безопасности может заподозрить подготовку атаки.

Другие угрозы выражаются в четкой последовательности действий и имеют формализованное воплощение в виде **эксплойта** (exploit) — программы или просто последовательности командных строк, некоторой порции данных и/или пошаговом описании действий, которые, будучи выполненными, позволяют злоумышленнику воспользоваться некоторой конкретной уязвимостью ИС в своих интересах. Особая опасность эксплойта состоит в том, что, имея его в своем распоряжении, даже малоподготовленный хакер способен провести успешную атаку. Для этого ему достаточно зайти на один из многочисленных сайтов, снабжающих всех желающих своей «продукцией». Более того, в придачу к инструкциям и программам в интернете можно найти даже предложения о сдаче в аренду целых **ботсетей**, готовых к реализации мощных кибер-атак. В то же время наличие у эксплойтов фиксированных признаков, таких, например, как специфические кодовые последовательности, облегчают распознавание и отражение соответствующих атак.

Угрозы могут исходить как от легальных пользователей сети, так и от внешних злоумышленников. Примерно 2/3 от общего числа всех наиболее серьезных инцидентов, связанных с безопасностью, составляют нарушения или ошибки легальных пользователей сетей: сотрудников и клиентов предприятий, студентов, имеющих доступ к сети учебного заведения, и др.

Угрозы со стороны легальных пользователей могут быть как умышленными, так и неумышленными. К *умышленным* угрозам относятся, например, доступ и похищение конфиденциальных данных, мониторинг системы с целью получения информации о ее устройстве, посещение запрещенных веб-сайтов, вынос за пределы предприятия съемных носителей и т. п.

Безопасность может быть нарушена и в результате *непреднамеренных* нарушений пользователей и обслуживающего персонала — ошибок, приводящих к повреждению сетевых устройств, данных, программного обеспечения, ОС и приложений, беспечность в сохранении секретности паролей и др. Известно, что правильное конфигурирование устройств является одним из мощных средств обеспечения безопасности. Но, будучи выполненной с ошибками, эта операция способна обернуться своей противоположностью — угрозой. Как выяснилось, некоторые «атаки» на ИС были на самом деле не атаками, а ошибками администраторов сетей при выполнении конфигурирования элементов системы. Например, широко известен случай неверного конфигурирования протокола маршрутизации BGP в сети клиента провайдера AS7007, который привел к отказам работы большой части интернета в 1997 году[1].

Угрозы внешних злоумышленников **(хакеров)** по определению являются умышленными и обычно квалифицируются как преступления. Отметим, что среди внешних нарушителей

[1] https://en.wikipedia.org/wiki/AS_7007_incident.

безопасности встречаются люди, занимающиеся этой деятельностью как профессионально, так и из хулиганских побуждений.

Ущерб и риск. Управление рисками

Известно, что абсолютная безопасность ИС не может быть обеспечена никакими средствами: всегда есть вероятность появления ошибок и проведения новых атак со стороны злоумышленников. Поэтому целью обеспечения информационной безопасности является не исключение, а *минимизация* возможного негативного влияния, которое могут оказать на ИС существующие угрозы. Из этого также следует, что надо каким-то образом ранжировать угрозы, чтобы решить, какими из них можно пренебречь, а на какие обратить основное внимание. Естественной мерой опасности атак и угроз является возможный ущерб, связанный с каждым из этих нарушений.

Ущерб (loss, impact) — это негативное влияние на систему, оказываемое проведенной атакой.

Подчеркнем, что в качестве ущерба рассматриваются не только и не столько потери, связанные с восстановлением работы ИС, в частности серверов, файловой системы или системы аутентификации, — главное внимание должно быть уделено потерям, которые в результате этих нарушений понесло предприятие, строящее свой бизнес на базе этой ИС.

Важнейшей задачей обеспечения информационной безопасности является управление рисками. Здесь **риск** определяется как оценка ущерба от атаки с учетом вероятностной природы атаки. Другими словами, риск характеризуется парой:

{Ущерб от атаки, Вероятность атаки}.

Суть **управления рисками** — системный анализ угроз, прогнозирование и оценка их последствий для предприятия, ранжирование угроз по степени их вероятного осуществления и опасности последствий и, наконец, выбор на приоритетной основе контрмер, направленных на смягчение или исключение возможного негативного воздействия этих нарушений на деятельность предприятия.

Управление рисками включает три укрупненных этапа (рис. 28.5):

1. Анализ уязвимостей.
2. Оценка рисков.
3. Управление рисками, или риск-менеджмент (принятие конкретных мер).

Анализ уязвимостей — объективное обследование реально существующих компьютерной сети, административных процедур и персонала. Угрозы определяются по отношению к *активам* предприятия, то есть ресурсам предприятия, представляющим для него ценность и являющимся объектом защиты (оборудование, недвижимость, транспортные средства, вычислительные устройства, ПО, документация и др.). Перечень угроз формулируется предположительно, то есть с использованием вероятностных категорий.

Оценка рисков — ранжирование возможных атак по степени опасности. Для этого вычисляются соответствующие риски — вероятностные оценки ущерба, который может быть нанесен предприятию каждой из атак в течение некоторого периода времени. Риск атаки тем выше, чем больше ущерб от нее и чем выше ее вероятность.

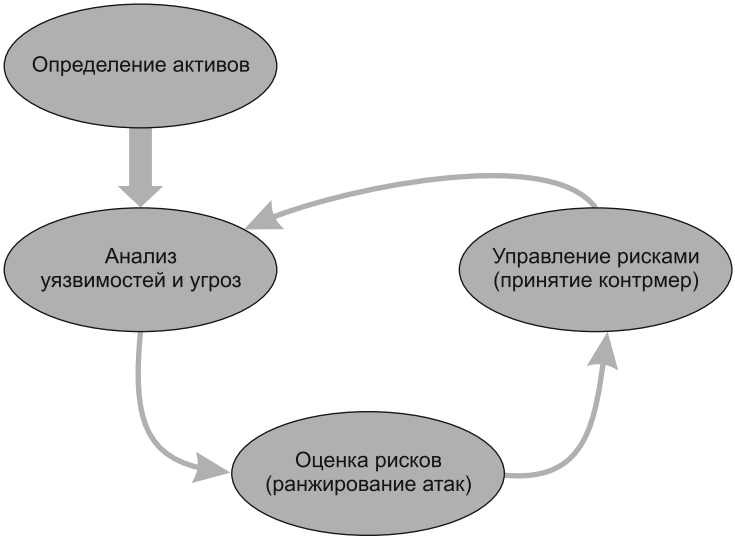


Рис. 28.5. Управление рисками

Риск-менеджмент — по каждому риску предпринимаются меры из следующего списка:

- *Принятие риска.* Этот вариант касается неизбежных атак, наносящих приемлемый ущерб.
- *Устранение риска.* Данный вариант имеет место, когда существующий риск можно свести на нет устранением либо уязвимости (например, сделать код коммерческого программного продукта открытым), либо угрозы (допустим, установить антивирусную систему).
- *Снижение риска.* Если риск невозможно ни принять, ни устранить, то предпринимаются действия по его снижению. Например, всегда существует некоторая вероятность проникновения злоумышленников в систему путем подбора паролей. В таком случае риск несанкционированного доступа можно снизить, установив более строгие требования к длине и сменяемости паролей.
- *Перенаправление риска.* Если риск невозможно ни принять, ни устранить, ни даже существенно снизить, то риск может быть перенаправлен страховой компании.

Типы и примеры атак

Пассивные и активные атаки

Атаки разделяют на активные и пассивные.

Активные атаки включают явные воздействия на систему, изменяющие ее состояние. Это могут быть зловредный программный код-вирус, внедренный в исполняемую системой программу, искажения данных на страницах взломанного веб-сайта, блокировка сетевого сервиса путем «бомбардировки» его ложными запросами или внедренное в коммуника-

ционный протокол ложное сообщение. Главной отличительной чертой активных атак является то, что после своего завершения они, как правило, оставляют следы.

Многие активные кибератаки относят к типу *взламывания* (breaking-in) по аналогии с бытовыми ограблениями со взломом, когда хозяин заходит в свой дом и сразу обнаруживает поврежденные замки, опустошенные ящики и разбросанные на полу вещи. В компьютерной системе после активного проникновения злоумышленника тоже остаются следы «взлома» — например, изменяется содержимое памяти, поступают странные диагностические сообщения, приложения начинают выполняться неправильно, замедленно или вообще зависают, в характеристиках сетевого трафика и в других статистических данных о работе системы появляются необъяснимые всплески активности.

Пассивные атаки не нарушают нормальной работы ИС. Они связаны со сбором сведений об ИС, например, прослушиванием внутрисетевого трафика или перехватом сообщений, передаваемых по линиям связи. Во многих случаях пассивные атаки не оставляют следов, поэтому их очень сложно выявить (часто они так и проходят незамеченными). Если использовать военную терминологию, то это разведка (но не боем).

Противопоставление активной и пассивной форм атаки является некоторой идеализацией. На практике мы редко имеем дело с активной или пассивной атакой «в чистом виде». Чаще всего атака включает подготовительный этап сбора информации об атакуемой системе, а затем на основе собранных данных осуществляется активное вмешательство в ее работу. К полезной для хакера информации относятся типы ОС и приложений, IP-адреса, номера портов, имена и пароли пользователей. Часть информации такого рода может быть получена при анализе открытой информации или простом общением с персоналом (это называют **социальным инжинирингом**), а часть — с помощью тех или иных программ. В последнем случае мы сталкиваемся с другой последовательностью этапов: сначала выполняется активная фаза внедрения на атакуемый компьютер подслушивающей программы, затем период пассивного сбора информации (например, паролей пользователей), а после этого — снова активная фаза проникновения в компьютер.

Отказ в обслуживании

К числу активных атак относятся две весьма распространенные атаки: отказ в обслуживании и распределенная атака отказа в обслуживании.

Смысл атаки **отказа в обслуживании** (Denial of Service, DoS) прямо следует из ее названия. Система, предназначенная для выполнения запросов легальных пользователей, вдруг перестает это делать или делает с большими задержками, что эквивалентно отказу. Очевидный пример такой системы — веб-сайт. Наверняка 17 млн британских болельщиков Энди Марри «обрушили» бы сайт BBC, если бы трансляция финального теннисного матча Уимблдона в 2013 году шла только в Интернете (к счастью, параллельно шла телевизионная передача). Такие всплески запросов являются экстраординарными, и правильно спроектированные серверы справляются с нагрузкой, на которую они рассчитаны. Однако отказ в обслуживании может наступить в результате не только резкой флюктуации интенсивности запросов, но и злонамеренных действий, когда перегрузка создается искусственно — допустим, когда на атакуемый компьютер посылается интенсивный поток запросов, сгенерированных средствами атакующего компьютера. Этот поток «затопляет» атакуемый компьютер, вызывая его перегрузку, и в конечном счете делает его не-

доступным. Блокировка происходит в результате исчерпания ресурсов либо процессора, либо операционной системы, либо канала связи (волосы пропускания).

Злоумышленник может многократно усилить эффект от проведения атаки отказа в обслуживании путем кражи чужой вычислительной мощности . Для этого он получает контроль над атакуемым компьютером, загружает в него вредительское программное обеспечение и активирует его. Таким образом, злоумышленник незаметно для владельца «ответвляет» часть вычислительной мощности, заставляя компьютер работать на себя. При этом владельцу компьютера не наносится никакого другого вреда, кроме снижения производительности его компьютера. Для проведения мощной атаки злоумышленник захватывает контроль над некоторым множеством компьютеров (рис. 28.6), организует их согласованную работу и направляет суммарный, многократно усилившийся поток запросов с множества компьютеров-зомби на компьютер-жертву. Говорят, что в таких случаях имеет место **распределенная атака отказа в обслуживании** (Distributed Denial of Service, DDoS), или DDoS-атака.

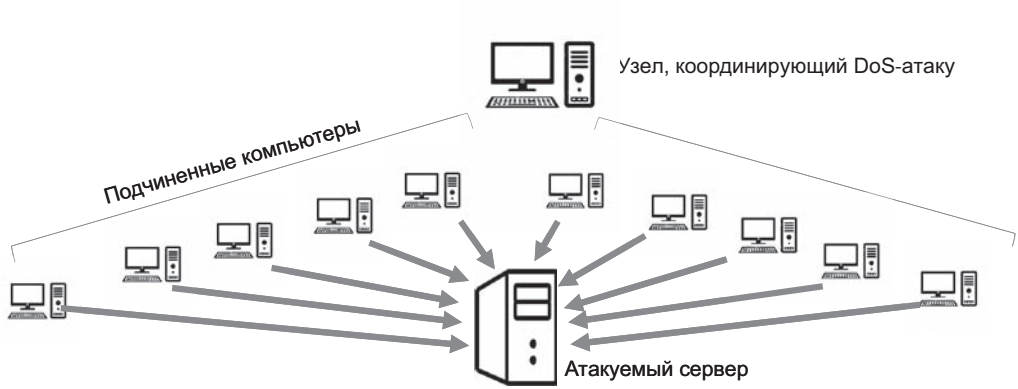


Рис. 28.6. Схема DDoS-атаки

При проведении атак злоумышленнику важно не только добиться своей цели, заключающейся в причинении ущерба атакуемому объекту, но и уничтожить все следы своего участия в этом. Одним из основных приемов, используемых злоумышленниками для «заметания следов», является *подмена содержимого пакетов*, или **спуфинг** (*spoofing*). В частности, для сокрытия места нахождения источника вредительских пакетов (например, при атаке отказа в обслуживании) злоумышленник изменяет значение поля адреса отправителя в заголовках пакетов. Поскольку адрес отправителя генерируется автоматически системным программным обеспечением, злоумышленник вносит изменения в соответствующие программные модули так, чтобы они давали ему возможность отправлять со своего компьютера пакеты с любыми IP-адресами. Еще труднее определить адрес источника распределенной атаки, так как непосредственными исполнителями выступают «зомбированные» компьютеры и именно их адреса содержатся в поле адреса отправителя пакетов, «бомбардирующих» компьютер-жертву. И хотя ничего не подозревающие владельцы компьютеров-исполнителей становятся участниками распределенной атаки помимо своей воли, большая часть ответственности ложится и на них. Ведь именно их недоработки в деле обеспечения безопасности собственных систем сделали возможной эту атаку.

Внедрение вредоносных программ

Многочисленная группа активных атак связана с внедрением в компьютеры **вредоносных программ** (malware — сокращение от malicious software). К этому типу программ относятся троянские и шпионские программы, руткиты, черви, вирусы, спам, логические бомбы и др. (рис. 28.7).

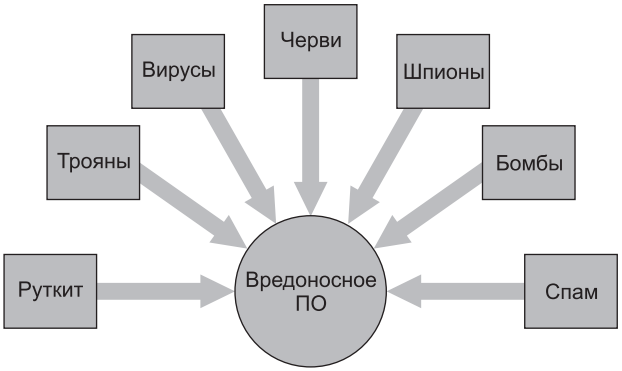


Рис. 28.7. Вредоносные программы

Эти программы могут проникать на атакуемые компьютеры разными путями. Самый простой из них — «самодоставка», когда пользователь загружает файлы из непроверенных источников (съемных носителей или веб-сайтов) либо беспечно открывает подозрительный файл, пришедший к нему как приложение по электронной почте. Существуют и более сложные представители вредоносных программ, обладающие собственными механизмами «размножения», копии таких программ распространяются по компьютерам сети без участия пользователей.

Одним из примеров вредоносных программ являются **шпионские программы** (spyware), которые тайно (как правило, удаленно) устанавливаются злоумышленниками на компьютеры ничего не подозревающих пользователей, чтобы отслеживать и фиксировать все их действия. В число таких действий может входить введение имени и пароля во время логического входа в систему, посещение тех или иных веб-сайтов, обмен информацией с внешними и внутренними пользователями сети и пр. Собранная информация пересылается злоумышленнику, который применяет ее в преступных целях.

Заметим, что в качестве шпионских программ могут использоваться не только созданные специально для этих целей вредоносные программы, но и программы легального назначения. Так, опасным средством шпионажа могут стать легальные системы мониторинга сети, такие, например, как популярные сетевые мониторы Wireshark или Microsoft Network Monitor. Исходное назначение этих программ состоит в том, чтобы дать администратору сети возможность следить за сетевым трафиком, в частности, захватывать пакеты, используя механизм фильтрации, просматривать их содержимое, собирать статистику по загрузке устройств. В руках злоумышленника такая программа превращается в мощный инструмент взлома сети, который позволяет перехватывать пакеты с паролями и другой секретной информацией. Потери, вызванные вредоносными программами, могут заключаться не только в уничтожении, искажении или похищении информации, приведении в нерабочее

состояние программного обеспечения, а значит, и компьютера в целом, но и в значительных затратах времени и сил администраторов на обнаружение и распознавание атак, фильтрацию внешних сообщений, тестирование и перезагрузку систем.

Кража личности, фишинг

По мере развития услуг, оказываемых через интернет, все более популярными становятся аферы, когда один человек выдает себя за другого. Действительно, ведь в этом случае не требуется личное присутствие в офисе, и индивидуум доказывает свою идентичность, передавая обслуживающему центру свои персональные данные по телефону или используя интерактивную систему веб-сайта. Злоумышленник решает выдать себя за другого, чтобы, например, взять кредит на чужое имя, получить доступ к чужому счету, рассчитаться за покупку чужой карточкой, получить именное приглашение на закрытое мероприятие. Такую *пассивную* атаку, заключающуюся в сборе данных о другом человеке, называют **кражей личности** (identity theft).

Фишинг (phishing — искаженное fishing) используется мошенниками для «выуживания» персональных данных. К примеру, вы отвечаете на телефонный звонок, а человек, представившийся сотрудником банка или государственной налоговой службы, работником ЖКХ или представителем провайдера мобильной связи, начинает выспрашивать у вас персональные данные. Угроза может прийти и по электронной почте. Будущая жертва получает сообщение, в котором, к примеру, говорится о якобы произведенной ею покупке, как правило, достаточно дорогой. Далее говорится, что при снятии средств за эту покупку у банковской системы возникли некие проблемы. Для разрешения ситуации клиенту предлагается срочно пройти по ссылке на сайт банка. Жертва, взволнованная тем, что никакой такой покупки она не совершала, торопится прояснить ситуацию, щелкает на предложенной ссылке и видит на экране знакомый логотип своего банка и интерактивную форму, запрашивающую персональные данные клиента, ИНН, номер счета, девичью фамилию матери и другие данные, которые нужны злоумышленнику.

Чем больше людей узнает о приемах выуживания информации, тем более изощренные методы обмана применяют преступники. Они создают поддельные сайты, выглядящие как настоящие, и используют доменные имена, очень похожие на настоящие. К примеру, когда вам предлагают посетить сайт международной платежной системы PayPal, а в адресной строке браузера появляется адрес www.pеypal.com, вы можете и не заметить подмены. Когда же наученные горьким опытом пользователи интернета стали более внимательными, мошенники научились, используя несовершенства браузеров, помещать в поле адресной строки браузера имя настоящего сайта, в нашем случае — paypal.com. В такой ситуации даже самый внимательный пользователь может потерять бдительность и перейти на подставной сайт. И хотя эта уязвимость браузера была вскоре устранена, расслабляться нельзя — преступники продолжают совершенствовать приемы фишинга.

Следующим изобретением стали всплывающие окна. Предположим, клиент получает доступ к сайту своего банка (действительному, не поддельному) в результате прохождения стандартной процедуры идентификации и аутентификации. Он просматривает страницы сайта, причем у него нет никаких сомнений в том, что это реальный сайт. В какой-то момент на экране появляется всплывающее окно, которое стилистически выглядит как неотъемлемая часть сайта. В этом окне размещена интерактивная форма, запрашивающая персональные данные. Клиент чувствует себя в полной безопасности и вводит все запра-

шиваемые данные. Однако в действительности эти всплывающие окна не являются частью настоящего сайта банка, а создаются зараженным браузером.

Иерархия средств защиты

Обычно первое, что ассоциируется с информационной безопасностью, — это антивирусные программы, файерволы, системы шифрования, аутентификации, аудита и другие технические средства защиты. Бесспорно, роль этих средств в обеспечении безопасности велика, однако не меньшее, а иногда и большее влияние на безопасность системы оказывают средства, построенные на качественно иной основе.

Видеокамера и надежный замок в офисе, продуманная процедура приема сотрудников на работу, закон, угрожающий хакеру уголовным преследованием, стандарт, помогающий провести анализ возможного ущерба из-за действия нарушителя, — все эти мало схожие между собой средства одинаково важны для обеспечения безопасности.

Успех в области информационной безопасности может принести только *системный подход*, при котором средства защиты разных типов применяются совместно и под централизованным управлением.

Общепризнанным является представление множества разных средств защиты в виде четырех иерархически организованных уровней: законодательного, административного, процедурного и технического уровней. Средства каждого из них могут быть использованы на разных этапах жизненного цикла системы обеспечения информационной безопасности.

Средства безопасности законодательного уровня. К этому уровню средств безопасности относятся правовое регулирование, стандартизация, лицензирование и морально-этические нормы, принятые в обществе. Законодательство может прямо влиять на концепцию построения защиты. Например, выход Федерального закона «О персональных данных», регламентирующего меры по обеспечению безопасности персональных данных при их обработке, потребовал от многих предприятий пересмотра и внесения принципиальных изменений в процедуры и инфраструктуру обработки информации. Важным направлением законодательства в области безопасности является и стандартизация. Стандартные процедуры оценки систем дают возможность их сопоставления и сравнения, на основании результатов которых может выполняться **сертификация систем** на соответствие определенным требованиям. К числу самых известных сертификационных стандартов относят **Оранжевую книгу**[1]. Этот самый заслуженный и популярный стандарт оценивает степень защищенности ОС.

Средства безопасности административного уровня базируются на *политике безопасности*, которая определяет стратегические направления информационной защиты предприятия — очерчивает круг критически важных информационных ресурсов предприятия, защита которых представляет наивысший приоритет, предлагает возможные меры устранения или уменьшения связанных с этими ресурсами рисков. На основе найденной стратегии разрабатывается программа обеспечения безопасности ИС, планируется совокупный бюджет, необходимый для выполнения программы, назначаются руководители и очерчивается зона их ответственности.

---

[1] Формальное название этого стандарта: «Министерство обороны США, Критерии оценки доверенных компьютерных систем» (Department of Defence, Trusted Computer System Evaluation Criteria).

Средства безопасности процедурного уровня решают задачи, поставленные вышележащим административным уровнем, с использованием технических средств, предоставляемых нижележащим техническим уровнем. В качестве основного средства процедурного уровня выступает человек, выполняющий взаимосвязанную последовательность действий, направленную на решение той или иной задачи обеспечения безопасности. Любой аспект информационной безопасности предполагает использование средств процедурного уровня. Даже простое поддержание нормального режима работы ИС осуществляется за счет выполнения множества повседневных процедур: резервного копирования, управления программным обеспечением, профилактических работ и т. п. К средствам процедурного уровня относится также управление персоналом, пропускной режим на территорию предприятия, охрана границ территории и др.

Средства безопасности технического уровня можно разделить на программные, аппаратные и программно-аппаратные.

Программные средства включают защитные инструменты операционных систем (подсистемы аутентификации и авторизации пользователей, средства управления доступом, аудит и др.) и прикладные программы, предназначенные для решения задач безопасности (системы обнаружения и предотвращения вторжений, антивирусные средства, прокси-серверы).

Примером аппаратных средств, специализирующихся на информационной защите, являются источники бесперебойного питания, генераторы напряжения, средства контроля доступа в помещения и пр.

К аппаратно-программным средствам относятся, например, некоторые анализаторы сетевого трафика и межсетевые экраны. И хотя данный уровень средств называется техническим, к нему также относят математические методы (методы криптографии), алгоритмы (эвристический алгоритм расчета времени оборота в протоколе TCP), абстрактные модели (модели контроля доступа) и т. п. Именно техническому уровню средств безопасности уделяется основное внимание в этой книге.

Принципы защиты информационной системы

Далее рассмотрены принципы построения системы обеспечения информационной безопасности, многие из которых имеют универсальный характер и применимы для защиты систем самой разной природы.

Подход сверху вниз

Подход «сверху вниз», где понятие «верх» означает руководство предприятия, а «низ» — уровень рядовых сотрудников, соответствует универсальному принципу движения от общего к частному. При таком подходе все принципиальные решения принимаются топ-менеджментом, затем руководители промежуточных уровней преобразуют их в более развернутые планы и частные решения, которые, наконец, доводятся в виде инструкций и в разной степени формализованных процедур до уровня исполнителей. Именно руководители предприятия определяют стратегически важные объекты защиты, оценивают риски,

которые может понести предприятие в результате разрушения тех или иных информационных активов, намечают стратегию защиты информационных ресурсов.

Противоположный подход — «от частного к общему», или «снизу вверх», успешно используемый в некоторых сферах деятельности (например, в научных исследованиях), совершенно неприменим для проектирования сложных технических систем, к которым относится система обеспечения безопасности. Решения, принимаемые на уровне специалистов отдельных подразделений, могут оказаться несогласованными и не способствовать достижению глобальной цели. Например, системный администратор на свой страх и риск, приложив большие усилия и потратив значительные средства, обеспечил надежную защиту базы данных, а в ней, как впоследствии оказалось, хранится легко восстанавливаемая информация, которая не представляет для бизнеса особой ценности.

Защита как процесс

Защита должна представлять собой непрерывный, циклический, проактивный процесс.

Задача информационной защиты не может быть решена раз и навсегда — напротив, работа по защите ИС должна идти непрерывно на протяжении всего существования защищаемой системы. Все системы безопасности уникальны, поскольку отражают специфику конкретных предприятий, для защиты которых они предназначены. Но какие бы различные цели ни преследовались при их создании, какие бы различные технологии ни использовались, какие бы индивидуальные решения ни принимались, общий ход проектирования для всех правильно построенных систем защиты должен иметь циклический характер. Цикл должен включать анализ угроз и уязвимостей защищаемой системы, оценку рисков, разработку политики безопасности всех уровней и реализацию принятых решений, направленных на снижение рисков.

Процесс обеспечения безопасности по возможности должен иметь *проактивный* (упреждающий), а не реактивный характер. При реактивном подходе защита заключается в принятии мер уже после того, когда нарушение безопасности произошло. Очевидно, что для успешности отражения атаки в первую очередь важны правильно выбранные действия и скорость их выполнения, а как раз этого трудно ожидать в ситуации кризиса. Поэтому более предпочтительным является проактивный подход, когда для защиты от вероятных угроз в спокойной обстановке проводится основательная подготовка оборонительных мер: устанавливаются необходимые технические средства, продумываются действия персонала, составляются и документируются инструкции — то есть делается все, что только может быть сделано заранее.

Эшелонированная защита

Эффективная защита обеспечивается путем многократного резервирования средств безопасности.

Надежность решения любой задачи повышается, если использовать резервирование. Задача обеспечения безопасности не является здесь исключением. Так, например, для обе-

спечения физической сохранности важного документа могут применяться самые разные средства защиты: дверные замки, датчики разбития окон, противопожарные сигнальные устройства, тревожная кнопка, сейф и масса других полезных приспособлений.

Информационная система существует в окружении гораздо более изощренных и многообразных угроз, здесь тем более невозможно найти панацею — одно-единственное средство, которое могло бы со стопроцентной надежностью противостоять всем видам атак. Поэтому на пути к защищаемому информационному ресурсу, как правило, устанавливают несколько барьеров. Вместе с тем возникает резонный вопрос: если ни одно из средств обеспечения безопасности не является абсолютно надежным и в принципе может быть преодолено злоумышленником, то в чем смысл нескольких защитных рубежей? Ответ: многократное резервирование в системах защиты служит не столько для того, чтобы какое-то из защитных средств продублировало отказавшее, а главным образом для того, чтобы заставить преступника *потратить как можно больше времени* на преодоление череды защитных барьеров. Замедление атаки повышает шанс ее обнаружения и принятия адекватных мер.

Рассмотрим, например, как реализуется принцип эшелонированной защиты в случае, когда необходимо обеспечить безопасность данных, хранящихся на одном из хостов внутренней локальной сети предприятия. На рис. 28.8 концентрическими окружностями представлены рубежи обороны, каждый из которых добавляет к уже накопленному защитному потенциалу собственные средства защиты (некоторые виды этих средств обеспечения безопасности рассмотрены в последующих главах).



Рис. 28.8. Рубежи обороны ИТ-системы

Самый внешний слой (организационно-административный) решает задачу безопасности данных, затрудняя злоумышленникам физический доступ к данным. С этой целью разрабатываются и применяются административные и организационные меры безопасности: проверка персонала при приеме на работу, взаимный контроль персонала, ограничение использования переносных портативных носителей и др.

Следующий слой также направлен на защиту от физического проникновения, но другими средствами — средствами физической защиты: ограждения, освещение, видеокамеры, контроль входа в здание, двери с кодовыми замками и т. п.

Далее вступают в действие технические средства безопасности, которые для сети с типовой структурой включают следующие рубежи защиты:

- внешняя сеть — для защиты от проникновения применяются средства регистрации входа, аудит, защитные свойства VPN;
- периметр внутренней сети — защита усиливается за счет файервола и прокси-серверов;
- внутренняя сеть — добавляются системы обнаружения и предотвращения вторжений сетевого уровня;
- хост — дополнительно проводятся процедуры аутентификации и авторизации, работают программный файервол, антивирус, системы обнаружения и предотвращения вторжений уровня хоста;
- данные — механизм контроля доступа и шифрование.

Сбалансированная защита

> Степень защищенности системы измеряется защищенностью ее самого слабого звена.

Этот принцип можно сформулировать и несколько по-другому: при построении системы безопасности необходимо обеспечить баланс стойкости всех ее компонентов. Например, если в сети все сообщения шифруются, но ключи легкодоступны, то эффект от шифрования окажется нулевым.

Из данного принципа можно сделать и еще одно заключение: если у злоумышленника существует несколько путей нанести урон системе и один из этих путей имеет слабую защиту, то нет смысла добиваться высокого качества защиты других путей. То есть если внешний трафик сети, подключенной к интернету, проходит через мощный сетевой экран, но пользователи имеют возможность связываться с узлами интернета по коммутируемым линиям через локально установленные модемы, то деньги (как правило, немалые), потраченные на сетевой экран, можно считать выброшенными на ветер. В таких случаях оказывается полезным еще один принцип — *принцип единого контрольно-пропускного пункта*, который заключается в том, что весь входящий во внутреннюю сеть и выходящий во внешнюю сеть трафик проходит через единственный узел сети, например межсетевой экран.

Необходимость баланса стойкости разных компонентов системы безопасности особенно ярко иллюстрируется провалами силовых ведомств, когда мощные организации, располагающие гигантскими ресурсами защиты, допускают существование явных прорех в своих системах безопасности. Так, известен случай, когда дисковый накопитель с классифицированными данными вооруженных сил США был случайно обнаружен продаю-

щимся на базаре в Ираке. Причина — использование ненадежных процедур утилизации данных и аппаратуры, хотя для остальных стадий существования данных — хранения и передачи — использовались мощные алгоритмы шифрования. Еще два примера связаны с масштабными утечками сверхсекретных данных — их главными действующими лицами были Мэннинг (2010 г.) и Сноуден (2013 г.). В обоих случаях очевидным слабым звеном стало управление персоналом: несмотря на то, что незадолго до инцидентов в поведении потенциальных нарушителей их коллегами отмечались «странности», а также то, что в карьере каждого из них произошли события, которые обычно квалифицируются как провоцирующие факторы, в отношении них не было предпринято никаких расследований. Кроме того, в обоих случаях не сработал и контроль использования портативных запоминающих устройств.

Компромиссы системы безопасности

Система обеспечения безопасности создается в результате компромисса между *качеством защиты*, с одной стороны, и *затратами* на разработку этой системы — с другой. Под качеством здесь понимается комплекс характеристик: функциональное разнообразие, надежность защиты, удобство работы сотрудников, поддерживающих систему безопасности, сотрудников других подразделений предприятия.

Пусть, например, на предприятии внедряется система защиты. Ее назначение — сократить прогнозируемый совокупный ущерб, который мог бы быть нанесен предприятию, если бы система защиты отсутствовала. При внедрении системы защиты возможный ущерб от атак снизился, однако в позиции «убытки» у предприятия добавились затраты на внедрение системы безопасности. Кроме того, к убыткам предприятия должны быть отнесены те потери, которые предприятие понесло из-за снижения производительности в результате внедрения системы безопасности. Подобное снижение может быть вызвано как дополнительными затратами вычислительных ресурсов, так и необходимостью выполнения сотрудниками предприятия дополнительных процедур, связанных с безопасностью.

Решение о внедрении системы безопасности можно считать экономически обоснованным только в том случае, если потери от риска в совокупности с затратами на систему безопасности и потерями из-за снижения производительности окажутся меньше исходного значения ущерба от риска.

Очевидно, что создание абсолютно непроницаемой защиты невозможно, так как у атакующих всегда остается теоретическая возможность взломать любую защиту — обычно это только вопрос времени и средств, которыми располагают злоумышленники. А это значит, что перед защищающейся стороной рано или поздно встанет дилемма: продолжать ли вкладывать деньги или остановиться на текущем уровне безопасности. Для ответа на этот вопрос разработчикам системы безопасности предлагается встать на место злоумышленника и попытаться оценить, какой уровень защиты злоумышленник мог бы посчитать неприемлемым для себя. Так, например, вряд ли имеет смысл браться за добычу конфиденциальных данных, если эта работа настолько длительная, что к тому времени, когда секретная информация попадет в руки, она уже устареет и не будет представлять никакой ценности. Аналогично, никто (из экономически мотивируемых преступников) не будет заниматься взломом системы, если выгоды от обладания защищаемым ресурсом меньше, чем средства, потраченные на проведение атаки. Исходя из этих соображений,

можно сформулировать следующие утверждения, каждое из которых представляет собой вариацию *принципа разумной достаточности*:

- Затраты на обеспечение безопасности информации должны быть, по крайней мере, не больше, чем величина потенциального ущерба от ее утраты.
- Стойкость системы безопасности считается достаточной, если время преодоления защиты превосходит время старения информации.
- Стойкость системы безопасности считается достаточной, если стоимость ее преодоления злоумышленниками превосходит стоимость полученной ими выгоды.

Таким образом, проектирование системы безопасности требует нахождения множества компромиссов между возможными затратами и возможными рисками. Так, в некоторых случаях можно отказаться от дорогостоящего файервола в пользу стандартных средств фильтрации обычного маршрутизатора, в других же приходится идти на беспрецедентные затраты.

Шифрование — базовая технология безопасности

Основные понятия и определения

Шифрование является краеугольным камнем всех служб информационной безопасности, будь то система аутентификации или авторизации, защищенный канал или средства безопасного хранения данных. Прежде чем перейти к конкретным методам и алгоритмам шифрования, давайте определим некоторые базовые понятия криптографии.

Шифрование — обратимое преобразование информации в целях обеспечения конфиденциальности данных. **Дешифрование** — процедура, которая, будучи примененной к зашифрованному тексту[1], снова приводит его в исходное состояние.

Пара процедур — шифрование и дешифрование — называется **криптосистемой**. Обычно криптосистема предусматривает наличие специального элемента — **секретного ключа**, в качестве которого может выступать некоторый предмет, например книга, число или рисунок. Простейший метод шифрования — замена букв в шифруемом тексте в соответствии с тем или иным правилом. Например, каждой букве алфавита может ставиться в соответствие другая буква этого алфавита, сдвинутая на некоторое число позиций влево или вправо. В качестве секретного ключа здесь выступает число, определяющее сдвиг.

Криптосистема считается *раскрытой*, если найдена процедура, позволяющая подобрать ключ за реальное время. Методы раскрытия криптосистемы, процедуры выявления уязвимости криптографических алгоритмов, выяснение секретного ключа называют **криптоанализом** или взломом шифра. Попытку раскрытия конкретного шифра с применением методов криптоанализа называют **криптографической атакой**.

Например, классическим методом криптоанализа, применяемым для раскрытия шифров, основанных на перестановке или замене букв, является частотный анализ. Для текстов, на-

[1] Информацию, над которой выполняются функции шифрования и дешифрования, мы будем условно называть «текстом», учитывая, что это может быть также числовой массив или графические данные.

писанных на определенном языке, относящихся к определенной сфере знаний, существуют устойчивые статистические данные о частоте, с которой встречается в тексте та или иная буква или последовательность букв, включая некоторые слова. Обладая такими данными и проведя статистический анализ зашифрованного текста, можно выполнить обратную замену символов.

Сложность алгоритма раскрытия является одной из важных характеристик криптосистемы и называется **криптостойкостью**. В криптографии принято **правило Керкгоффса**, заключающееся в том, что *стойкость шифра должна определяться только секретностью ключа*. Так, все стандартные алгоритмы шифрования (например, AES, DES, PGP) широко известны[1], их детальное описание содержится в легкодоступных документах, но от этого их эффективность не снижается. Система остается защищенной, даже если злоумышленнику известно все об алгоритме шифрования, но он не знает секретный ключ.

Существующие криптосистемы можно разделить на два класса — **симметричные** и **асимметричные**. В симметричных схемах шифрования (классическая криптография) секретный ключ шифрования совпадает с секретным ключом дешифрования. В асимметричных схемах шифрования (криптография с открытым ключом) ключ шифрования не совпадает с ключом дешифрования.

Симметричное шифрование

На рис. 28.9 приведена модель симметричной криптосистемы. В данной модели три участника: два абонента, желающих обмениваться шифрованными сообщениями, и злоумышленник, который хочет перехватить и каким-либо образом расшифровать передаваемые сообщения.

ПРИМЕЧАНИЕ

При объяснении алгоритмов шифрования здесь и далее мы будем называть участников обмена Алисой и Бобом, а злоумышленника, старающегося перехватить их сообщения, — Евой. Эти имена традиционно используются в криптографии.

В распоряжении Алисы и Боба имеется незащищенный канал передачи сообщений, который в принципе может прослушиваться злоумышленником. Поэтому они договариваются использовать шифрование и для этого им нужен секретный ключ, известный только им двоим. Этот ключ им был передан (или один из них послал его другому) заранее по другому каналу — надежному. Боб и Алиса, получив ключ, находятся в абсолютно равном (симметричном) положении, каждый из них может как посылать шифрованные сообщения, так и получать и расшифровывать их. Для определенности на рисунке показана схема передачи сообщений со стороны Боба.

Боб зашифровывает свое сообщение — открытый текст X — функцией шифрования F с секретным ключом k и передает в открытый канал результат — шифрованный текст Y. Алиса получает Y и передает его на вход функции дешифрования F', которая выполняет в обратном порядке все действия, выполненные ранее функцией F. Это может быть

[1] Вместе с тем существует немало фирменных алгоритмов, описание которых не публикуется для того, чтобы усилить защиту.

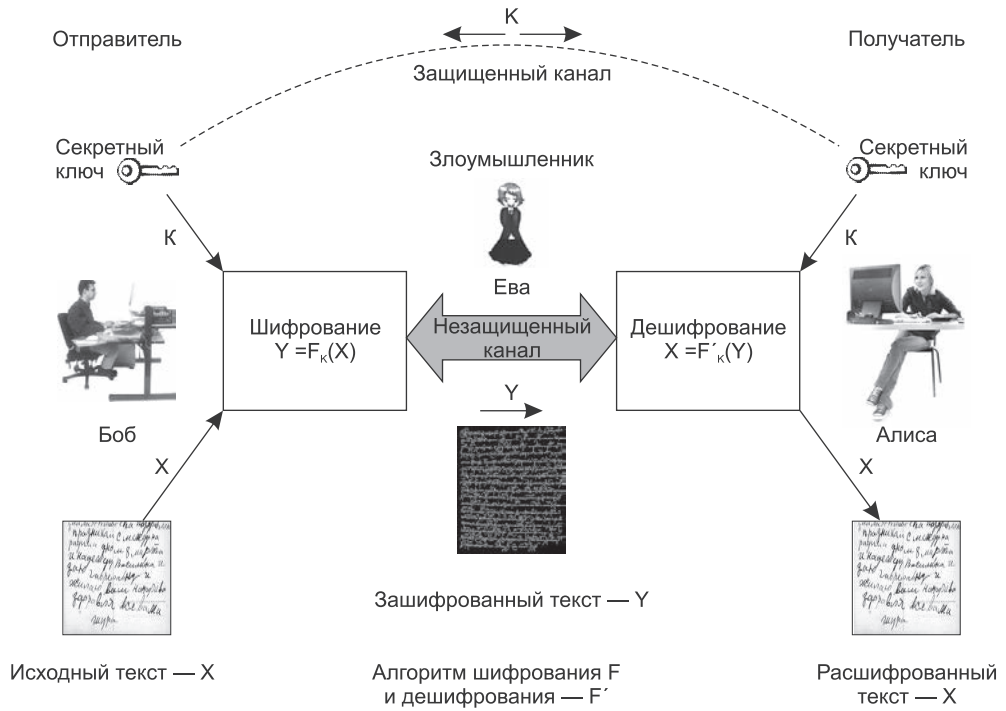


Рис. 28.9. Модель симметричного шифрования

сделано только в том случае, если на вход функции *F* будет подано то же самое значение параметра — значение ключа *k*. Алиса имеет секретный ключ и поэтому получает расшифрованное значение. При необходимости передавать шифрованные сообщения Бобу Алиса должна действовать аналогичным образом.

До недавнего времени наиболее популярным стандартным симметричным алгоритмом шифрования данных был **DES** (Data Encryption Standard). Для шифрования используется циклическая последовательность операций над битами шифруемого текста — перестановки, подстановки, логические операции двоичной арифметики. Эти операции применяются блочно, размер блока равен 64 битам. Некоторые операции над блоками шифруемых данных связаны со значениями секретного ключа, также имеющего длину 64 бита. Промежуточный результат шифрования складывается по модулю 2 (операция XOR) с преобразованной двоичной последовательностью ключа. Полный шифрованный текст получается слиянием результатов шифрования для всех блоков исходного текста.

Процедура дешифрования выполняется в обратном порядке. Поскольку собственно алгоритм DES не является секретом и широкодоступен, в том числе доступны все таблицы, описывающие перестановки, то стойкость алгоритма (степень сложности дешифрования) определяется только сложностью подбора ключа, которая прямо зависит от его длины.

Криптостойкость всех симметричных алгоритмов зависит от качества ключа, что предъявляет повышенные требования к службе генерации ключей, а также к надежности канала обмена секретными ключами между участниками секретных переговоров.

Чтобы повысить криптостойкость алгоритма DES, был разработан его усиленный вариант, называемый **тройным алгоритмом DES**, который включает троекратное шифрование с использованием двух разных ключей. При этом можно считать, что длина ключа увеличивается с 56 до 112 бит, а значит, криптостойкость алгоритма существенно повышается. Но за это приходится платить производительностью — тройной алгоритм DES требует в три раза больше времени на реализацию, чем «обычный».

В 2001 году был стандартизован симметричный алгоритм шифрования **AES** (Advanced Encryption Standard). AES обеспечивает лучшую защиту, так как использует 128-битные ключи (а также может работать со 192- и 256-битными ключами) и имеет более высокую скорость работы, кодируя за один цикл 128-битный блок, в отличие от 64-битного блока DES. В настоящее время, помимо AES, распространенным симметричным алгоритмом шифрования является алгоритм Blowfish. Утвержденные в качестве государственного стандарта РФ шифры «Магма» и «Кузнечик» также являются симметричными блочными методами шифрования с ключом 256 бит, при этом в первом из них используется блок размером 64 бита, а во втором — 128 бит.

(S) *Симметричный блочный алгоритм шифрования, использующий преобразования Фейстеля*

Проблема распределения ключей

Симметричный подход к шифрованию изначально несет в себе очевидную проблему, называемую проблемой **распределения ключей** (key distribution), которая состоит в следующем. Отправитель и получатель хотят обмениваться секретными сообщениями, но в их распоряжении имеется незащищенный открытый канал. Поэтому они вынуждены использовать шифрование, но чтобы послать зашифрованное сообщение, нужно предварительно обменяться секретной информацией о значении ключа. Однако секретный ключ нельзя передать по открытому каналу. Если его зашифровать другим ключом, то опять возникает проблема доставки второго ключа. Получается замкнутый круг.

Единственным по-настоящему надежным решением этой проблемы является передача ключа при личной встрече абонентов. Однако при активном обмене требуется часто менять ключи, чтобы не дать возможности криптоаналитику собрать большое количество шифрованного материала — известно, что чем больше зашифрованных сообщений окажется в руках криптоаналитика, тем легче ему раскрыть криптосистему. Кроме того, если злоумышленник перехватывает и сохраняет сообщения, зашифрованные одним и тем же ключом, то при раскрытии данного ключа они *все* окажутся скомпрометированными. Следовательно, необходимы частые личные встречи абонентов для обмена ключами, что, во-первых, не всегда возможно, а во-вторых, вообще делает бессмысленным обмен данными по каналу связи — действительно, зачем шифровать данные, если их можно лично передать при встрече.

Менее надежным способом распределения ключей является использование курьеров или других вариантов защищенной доставки ключей, но это решение тоже имеет очевидные изъяны. Существуют и другие приемы, не решающие, но смягчающие проблему распределения ключей. Например, у абонента может быть несколько секретных ключей, имеющих разное назначение. Один ключ выдается ему на долгий срок. Этот ключ применяется только для шифрования (дешифрования) других ключей — кратковременных, каждый из которых действителен только на время одного сеанса связи. И хотя в этом случае все равно

остается проблема доставки долговременного ключа, уже нет необходимости его частой смены, так как этот ключ используется относительно редко и шифрует небольшие порции данных — сеансовые ключи.

Несмотря на различные усовершенствования процедуры распределения ключей, они не могут полностью устранить коренной изъян симметричных методов — *необходимость доставки секретного ключа по незащищенному каналу*.

Если проблема с ключами возникает в системе с двумя абонентами, то она многократно усугубляется в системе с большим числом абонентов. Пусть, например, n абонентов желают обмениваться секретными данными по принципу «каждый с каждым» — значит, в этом случае потребуется $n(n-1)/2$ ключей и все они должны быть сгенерированы и распределены надежным образом. То есть *количество требуемых ключей пропорционально квадрату количества абонентов*, что при большом числе абонентов делает задачу чрезвычайно сложной. Но именно такая ситуация наблюдается во всех современных сетях связи — телефонных, радио и компьютерных. Все это сделало проблему распределения ключей чрезвычайно актуальной.

Метод Диффи — Хеллмана передачи секретного ключа по незащищенному каналу

В середине 70-х годов американские ученые Мартин Хеллман и Уилтфилд Диффи нашли способ, с помощью которого абоненты могли безопасно обмениваться секретными ключами без передачи их по каналу связи. Особенность этого открытия состоит в том, что оно противоречит всем интуитивным представлениям человека, делая возможным то, что кажется «очевидно» невозможным.

Метод Диффи — Хеллмана основан на использовании свойств односторонних функций.

Односторонняя функция (one-way function) — это функция $y = F(x)$, которая легко вычисляется для любого входного значения x, но обратная задача — определение x по заданному значению функции y — решается очень трудно. Примером односторонней функции может служить простейшая функция двух аргументов $F(p,q) = pq$, представляющая собой произведение двух простых чисел p и q, она вычисляется сравнительно просто, даже если числа p и q очень большие. Но чрезвычайно сложно решить обратную задачу (называемую факторизацией) — по произведению подобрать исходные два простых числа. Другой пример — функция $Y(x) = D^x \bmod P$, которая при некоторых ограничениях на параметры D и P является односторонней, то есть, зная Y, а также параметры D и P, нельзя без экстраординарных вычислительных усилий найти аргумент x.

Итак, пусть Алиса и Боб решили обмениваться шифрованными сообщениями, но в их распоряжении имеется только незащищенный открытый канал связи, при этом никаких возможностей встретиться или передать секретный ключ через кого-нибудь другого у них нет. В соответствии с алгоритмом Диффи — Хеллмана для успешного решения задачи Алиса и Боб должны выполнить следующие действия. Прежде всего они *открыто* договариваются о том, что будут использовать одностороннюю функцию $Y = D^x \bmod P$. Затем они договариваются о значениях параметров D и P. Пусть, например, они договорились, что $D = 7$ и $P = 13$, то есть функция имеет вид $Y = 7^x \bmod 13$. Еще раз подчеркнем, что в соответствии с алгоритмом Диффи — Хеллмана вся эта информация не является секретной, и даже если переговоры будут подслушаны Евой, это не даст ей возможности прочитать сообщения Алисы и Боба. Дальнейшие действия участников обмена описываются в табл. 28.1.

Таблица 28.1. Действия Алисы и Боба в соответствии с алгоритмом Диффи—Хеллмана

	Действия Алисы		**Действия Боба**	
1	Алиса секретным образом выбирает произвольное число A (закрытый ключ Алисы)	Пусть, например, $A = 2$	Боб также секретно выбирает произвольное число B (закрытый ключ Боба)	Пусть, например, $B = 4$
2	Алиса вычисляет значение a односторонней функции Y, используя в качестве аргумента свое секретное число A: то есть $a = D^A \bmod P$ (открытый ключ Алисы)	$a = 7^2 \bmod 13 = 10$	Боб также вычисляет значение b односторонней функции Y, используя в качестве аргумента свое секретное число B: $b = D^B \bmod P$ (открытый ключ Боба)	$b = 7^4 \bmod 13 = 2401 \bmod 13 = 9$
3	Алиса посылает Бобу свой открытый ключ a		Боб посылает Алисе свой открытый ключ b	
4	Алиса, получив от Боба число b, вычисляет по формуле $K = b^A \bmod P$ (разделяемый секретный ключ)	$K = 9^2 \bmod 13 = 81 \bmod 13 = 3$	Боб, получив от Алисы число a, вычисляет по формуле $K = a^B \bmod P$ (разделяемый секретный ключ)	$K = 10^4 \bmod 13 = 10000 \bmod 13 = 3$
5	По правилам модульной арифметики $b^A \bmod P = (D^B \bmod P)A \bmod P = D^{BA} \bmod P$	$K = 3$	По правилам модульной арифметики $a^B \bmod P = (D^A \bmod P)\, B \bmod P = D^{AB} \bmod P$	$K = 3$

В результате описанной процедуры на шаге 4 Алиса и Боб получили одно и то же число 3! Математические преобразования показывают, что вычисления Алисы и Боба всегда будут давать одинаковые результаты. Полученные в результате числа они могут использовать в качестве известного только им ключа для различных симметричных методов шифрования.

Посмотрим, может ли Ева подобрать разделяемый секретный ключ Алисы и Боба. Пусть на шаге 3, когда Алиса и Боб посылали друг другу свои открытые ключи a (10) и b (9), Ева смогла перехватить эти числа (ведь канал является открытым) и теперь пытается вычислить разделяемый секретный ключ. Зная число a, которое Алиса послала Бобу, Ева хочет повторить действия Боба и вычислить разделяемый секретный ключ по формуле $10^B \bmod 13$. Для этого ей требуется закрытый ключ Боба B, который он, однако, хранит секретно от всех. Зато Ева знает, что Боб использовал свой закрытый ключ B, когда вычислял значение своего открытого ключа — b. То есть задача будет решена, если Ева сможет подобрать такое значение B, чтобы значение $7^B \bmod 13$ равнялось 9. Но именно это практически неразрешимо, поскольку функция $7^B \bmod 13$ является односторонней. Таким образом, Алиса и Боб действительно получили секретный ключ.

Для того чтобы усложнить решение обратной задачи, то есть для восстановления закрытого ключа Алисы или Боба по открытому, на параметры алгоритма накладываются некоторые ограничения, в том числе следующие:

- все параметры D, P, A, B должны быть целыми положительными числами;
- A и B должны быть большими числами порядка 10^{100};
- P должно быть большим простым числом порядка 10^{300}, причем желательно, чтобы $(P - 1)/2$ также было простым числом;
- число D не обязательно должно быть большим, обычно оно выбирается меньше десяти, $D < P$.

Хотя алгоритм Диффи—Хеллмана стал прорывом в области криптографии, в его исходном состоянии он представлял скорее теоретическую, нежели практическую ценность. Устранив препятствие в виде необходимости надежного закрытого канала для передачи ключа, этот метод не снял проблемы квадратичной зависимости числа ключей от числа абонентов. Решение пришло очень скоро — уже через год после появления алгоритма Диффи—Хеллмана была теоретически доказана возможность принципиально нового подхода к шифрованию — асимметричного шифрования, при использовании которого (помимо прочих преимуществ) кардинально упрощается задача распределения ключей.

Концепция асимметричного шифрования

До сравнительно недавнего времени понятие «симметричное шифрование» не существовало просто потому, что все методы, которые использовались человечеством на протяжении нескольких тысяч лет, по современной классификации могли быть отнесены к классу симметричных, а других просто не было. Более того, все эти тысячи лет существовала твердая убежденность, что в принципе никогда не может быть иных схем, кроме симметричной, когда отправитель шифрует сообщение с помощью секретного ключа, а получатель с помощью этого же ключа сообщение расшифровывает!

Революция свершилась в конце 60-х — середине 70-х, когда с разницей в несколько лет две группы ученых, одна из которых — уже знакомые нам Диффи и Хеллман, а другая — сотрудники секретной правительственной лаборатории Великобритании[1] Эллис, Кокс и Уильямсон, независимо друг от друга изобрели принципиально новый подход к шифрованию, открывающий глобальные перспективы в области современных коммуникаций. Предельно упрощая, этот подход можно описать фразой: «отправитель шифрует сообщение с помощью одного ключа, а получатель расшифровывает его с помощью другого ключа». Как видим, здесь на двух сторонах обменного канала используются разные ключи, то есть присутствует асимметрия — соответственно все методы, основанные на таком подходе, стали называть «асимметричными».

Конечно, удивительно, что за несколько тысяч лет не было ни одной известной науке попытки изобретения асимметричного метода шифрования, и вдруг, практически одновременно, две независимые группы ученых совершают это открытие! Возможно, причина кроется в том, что к концу 60-х годов совпало два обстоятельства: во-первых, возникла острая потребность в новом типе шифрования, а во-вторых, появились технические возможности реализации этой идеи.

Потребность была продиктована зрелостью таких видов массовых коммуникаций, как телефон, радио, компьютерные сети, для которых, во-первых, особенно важна секретность ввиду слабой защищенности публичных средств связи, а во-вторых, неприемлемы ограничения традиционных методов шифрования, выражающихся в необходимости обмена секретным ключом для каждой пары абонентов. К концу 60-х годов стали отчетливо вырисовываться перспективы использования интернета как мировой сети связи, и одновременно с этим стало приходить осознание того, что глобальная публичная сеть может выполнить свою

[1] Известно, что исторически первыми были британские криптографы, которые открыли асимметричное шифрование на 6 лет раньше, чем Диффи и Хеллман, однако до 1997 года они не могли обнародовать свои результаты, так как их работа имела гриф секретности.

миссию только в том случае, если миллионам ее пользователей будет предоставлена возможность защищенного обмена сообщениями.

Эти темы особенно волновали военных разных стран, которых очень привлекала возможность распределенного управления вооруженными силами, но пугала невозможность гарантировать секретность передаваемых директив. И если в недалеком прошлом проблема распределения секретных ключей хотя и существовала, но была преодолимой, то в новых условиях она стала принципиальным препятствием.

К этому времени созрели технические возможности реализации вычислительно емких алгоритмов шифрования, к которым могут быть отнесены асимметричные алгоритмы. Массовое распространение получили компьютеры, обладающие такой вычислительной мощностью, которой до сих пор могли похвастаться только уникальные модели суперкомпьютеров. Это сделало шифрование обыденной операцией, которая может быть выполнена на обычном персональном компьютере.

Вот на таком историческом фоне и была предложена концепция асимметричной криптосистемы, называемой также **шифрованием с открытым ключом**.

На рис. 28.10 представлена модель асимметричной криптосистемы. Так же, как и в модели симметричного шифрования (см. рис. 28.9), здесь показаны три участника: отправитель (Боб), получатель (Алиса) и злоумышленник (Ева). В отличие от симметричной схемы шифрования, в которой наличие разделяемого секретного ключа автоматически означает возможность двустороннего защищенного обмена, здесь существует отдельная процедура для передачи зашифрованных сообщений в каждую из сторон. На рисунке показан вариант,

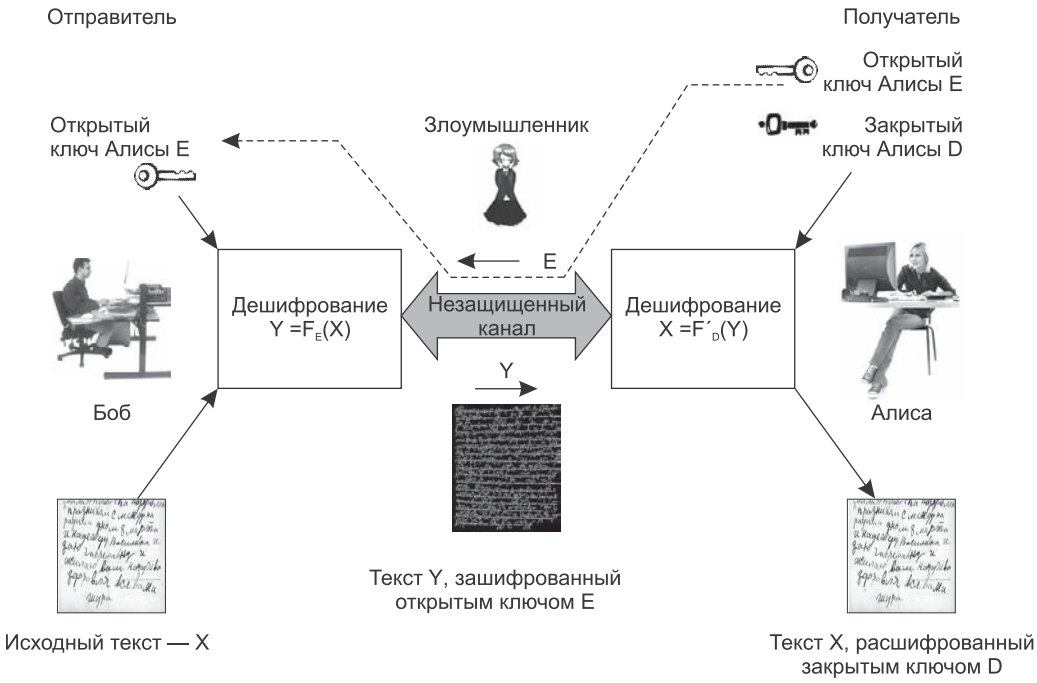


Рис. 28.10. Схема асимметричного шифрования

когда зашифрованные сообщения могут быть посланы только Бобом в сторону Алисы, но не наоборот.

1. Итак, Алиса пожелала, чтобы Боб посылал ей зашифрованные сообщения. Для этого она сгенерировала пару ключей: **открытый ключ** (public key) E и **закрытый ключ** (private key) D. Для шифрования текста служит открытый ключ, но расшифровать этот текст можно только с помощью закрытого ключа. Алиса не хочет, чтобы кто-либо читал ее почту, поэтому она сохраняет закрытый ключ D (часто называемый также личным ключом) в секрете. Открытый же ключ E Алиса свободно передает всем, от кого хочет получать зашифрованные сообщения. Открытый ключ не представляет никакого секрета, Алиса может поместить его на своей странице в социальной сети или обнародовать в рекламе на телевидении. Все, кто хотят посылать Алисе зашифрованные сообщения, используют один и тот же ключ E, но при этом никто из них не может прочитать сообщения друг друга.
2. Алиса передает Бобу свой открытый ключ E по незащищенному каналу в незашифрованном виде.
3. Боб шифрует свое сообщение X открытым ключом Алисы E и посылает зашифрованный текст $Y = F_E(X)$ по открытому каналу. Никто не может прочитать это сообщение. Даже сам Боб, если бы ему вдруг захотелось перечитать, что он там написал, не смог бы этого сделать, потому что для этого нужен закрытый ключ Алисы, которого у него нет.
4. Алиса получает шифрованное сообщение $Y = F_E(X)$ и расшифровывает его своим закрытым ключом D: $X = F'_D(Y)$.

Для того чтобы в сети все n абонентов имели возможность не только принимать зашифрованные сообщения, но и сами посылать таковые, каждый абонент должен обладать собственной парой ключей E и D. Всего в сети будет $2n$ ключей: n открытых ключей для шифрования и n секретных ключей для дешифрования. Таким образом решается проблема масштабируемости: *квадратичная зависимость количества ключей от числа абонентов в симметричных алгоритмах заменяется линейной зависимостью в асимметричных алгоритмах*. Решается и проблема доставки ключа; поскольку теперь он не является секретом, его можно без опаски передавать по открытому каналу. А злоумышленнику нет смысла стремиться завладеть открытым ключом, поскольку это не дает возможности расшифровать текст или вычислить закрытый ключ.

Алгоритм асимметричного шифрования RSA

Открыватели асимметричного подхода к шифрованию показали *концептуальную возможность* существования функций, позволяющих построить криптографическую систему, в которой текст шифруется одним ключом, а расшифровывается другим. Они также обрисовали те перспективы, которые открывает этот подход в деле решения проблемы распределения ключей. Ими были сформулированы два принципиальных требования, которым должны удовлетворять функции асимметричной криптосистемы:

- зашифрованное сообщение должно быть результатом вычислений односторонней функции, чтобы никто не мог выполнить обратные преобразования и получить исходный текст;
- эта односторонняя функция должна быть сконструирована таким образом, чтобы у нее был некоторый секретный элемент, зная который получатель шифровки мог бы легко выполнить обратное преобразование.

Функции, которые удовлетворяют данным требованиям, назвали **односторонними функциями с потайным входом** (trapdoor function). Некоторое время ученым не удавалось найти функций, удовлетворяющих этим критериям, поэтому идея асимметричного шифрования не находила практического применения. Наконец, в 1978 году трое американских ученых, Ривест, Шамир и Адлеман, предложили долгожданный **алгоритм асимметричного шифрования RSA**, названный так по первым буквам их фамилий — Rivest, Shamir, Adleman. В табл. 26.2 описываются основные шаги алгоритма RSA.

Таблица 26.2. Последовательность действий участников обмена данными в соответствии с алгоритмом RSA

Действия Алисы и Боба	Числовой пример
Алиса произвольно выбирает два случайных простых числа P и Q. Они должны быть очень большими — от этого зависит стойкость алгоритма шифрования	В примере для простоты расчетов берутся очень маленькие числа. Пусть $P = 7$ и $Q = 13$
Алиса вычисляет два произведения: $N = PQ$ $M = (P - 1)(Q - 1)$	$N = 91$ $M = 6 \times 12 = 72$
Алиса выбирает случайное целое число E, меньшее M и не имеющее с ним общих сомножителей	$E = 5$
Пара (E, N) — это открытый ключ Алисы, который она передает всем, от кого хочет получать шифрованные сообщения. Алиса посылает Бобу и всем остальным, с кем она желает вести защищенную переписку, свой открытый ключ (E, N)	(5, 91)
Алиса находит D такое, что $DE = 1 \bmod M$. Пара (D, N) — это закрытый ключ Алисы, который она не показывает никому. С этого момента она готова получать зашифрованные сообщения от Боба	$D \times 5 = 1 \bmod 72$ D = 29 (это число легко находится подбором, если учитывать признаки делимости на 5)
Боб получил открытый ключ Алисы и так же, как все остальные, имеющие доступ к этому ключу, может посылать Алисе зашифрованные сообщения. Он представляет свое сообщение в любом цифровом формате и разбивает его на блоки X таким образом, чтобы $0 < X < N$	Пусть секретный текст, посылаемый Бобом, состоит из одного символа R, который в коде ASCII имеет значение 1010010, или 82 в десятичном коде
Боб шифрует сообщение X открытым ключом (E, N): $C = X^E \bmod N$ и посылает Алисе зашифрованное сообщение C	$C = 82^5 \bmod 91 = =\{82^3 \bmod 91 \times 82^2 \bmod 91\} \bmod 91 = 10$ Вычисление модуля от степени числа упрощается при использовании следующего правила: $(Y^{a+b+c}) \bmod P = (Y^a \bmod P \times Y^b \bmod P \times Y^c \bmod P) \bmod P$
Алиса получает сообщение C и расшифровывает его своим закрытым ключом (D, N): $X = C^D \bmod N$	$X = 10^{29} \bmod 91 = \{10^1 \bmod 91 \times 10^4 \bmod 91 \times 10^6 \bmod 91 \ldots\} \bmod 91$ (внутри фигурной скобки четыре раза повторяется последний сомножитель — $10^6 \bmod 91$) $10 \bmod 91 = 10$; $10^4 \bmod 91 = 81$; $10^6 \bmod 91 = 1$ $X = \{10 \times 81 \times 1\} \bmod 91 = 82$
Результат расшифровки $X = 82$ совпадает с исходным секретным сообщением	

Еве для того, чтобы прочитать перехваченное сообщение C, требуется закрытый ключ Алисы (D, N). Но в ее распоряжении имеется только открытый ключ (E, N). Теоретически,

зная открытый ключ, можно вычислить значение закрытого ключа. Однако необходимым промежуточным действием в этом преобразовании является нахождение простых чисел P и Q, для чего нужно разложить на простые множители очень большое число N, а это является чрезвычайно трудоемкой процедурой. Таким образом, здесь мы имеем дело с односторонней функцией $N = P \times Q$. Но для Алисы это же действие — разложение большого числа на два простых множителя — не представляет никакого труда, потому что она знает, как сконструировано это число N, она сама его вычислила, произвольно выбрав два сомножителя. Другими словами, Алисе известен «потайной вход» этой односторонней функции. Именно с огромной вычислительной сложностью разложения большого числа N на простые множители P и Q связана высокая криптостойкость алгоритма RSA.

Хотя информация об открытом ключе не является секретной, ее нужно защищать от *подлогов*, чтобы злоумышленник под именем легального пользователя не навязал свой открытый ключ, после чего с помощью своего закрытого ключа он мог бы расшифровывать все сообщения, посылаемые легальному пользователю, и отправлять свои сообщения от его имени. Решение проблемы дает технология **цифровых сертификатов**[1] — электронных документов, которые связывают конкретных пользователей с конкретными открытыми ключами.

Хеш-функции. Односторонние функции шифрования. Проверка целостности

В области информационной безопасности особое место занимает специальный класс односторонних функций, называемых хеш-функциями.

Хеш-функцией (hash function) называют одностороннюю функцию, которая, будучи примененной к некоторым данным, дает в результате значение, состоящее из фиксированного сравнительно небольшого и не зависящего от длины исходных данных числа байтов. Результат работы хеш-функции называют **хеш-кодом** или **дайджестом**. Рассмотрим, например, функцию взятия модуля $Y(x) = x \bmod n$, где операция $x \bmod n$ (то есть x по модулю n) дает в результате остаток от деления x на n. Эта функция, во-первых, является односторонней, так как, зная остаток от деления x на n, невозможно однозначно определить значение аргумента x, во-вторых, она относится к классу хеш-функций, поскольку ее результат не зависит от аргумента x и всегда находится в диапазоне от 0 до $(n - 1)$.

Хеш-функции называют также **односторонними функциям шифрования (ОФШ)**, где в качестве шифрованного представления исходных данных выступает дайджест. При этом знание дайджеста *не позволяет* и даже *не предполагает* восстановления исходных данных. Односторонние функции шифрования используют в разных целях, в том числе для обеспечения целостности и аутентичности информации. Пусть, например, требуется обеспечить целостность сообщения, передаваемого по сети. Отправитель и получатель договорились, что они будут использовать одностороннюю функцию H с секретным числом — ключом K — в качестве параметра. Прежде чем отправить сообщение X, отправитель вычисляет для него дайджест $M = H(X, K)$ и отправляет его вместе с сообщением X адресату (рис. 28.11). Адресат, получив данные X и M, применяет ту же самую ОФШ к переданному в открытом виде исходному сообщению X, используя известный ему секретный ключ K:

[1] См. раздел «Аутентификация на основе цифровых сертификатов» главы 29.

$M' = H(X, K)$. Если значения дайджестов вычисленного локально M' и полученного по сети M совпадают, то содержимое сообщения не было изменено во время передачи.

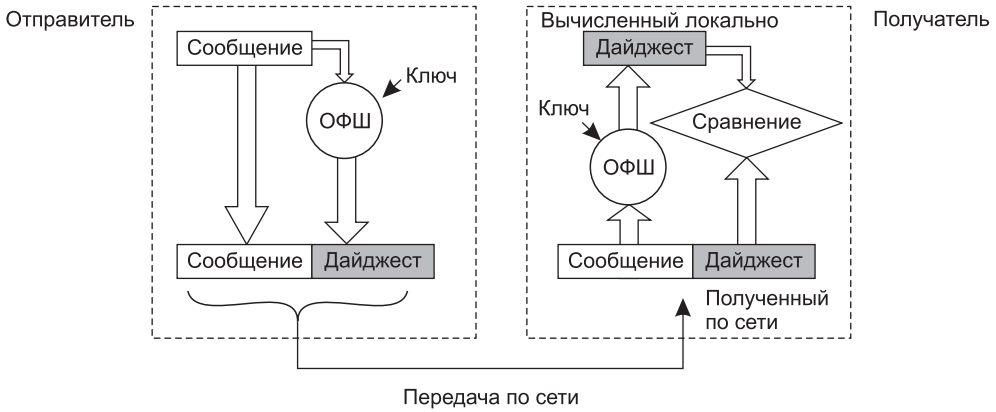


Рис. 28.11. Использование параметрической односторонней функций шифрования для контроля целостности

Хеш-функции широко используются в сетевых протоколах, алгоритмах электронно-цифровой подписи, механизмах аутентификации на основе паролей. Наиболее популярной в системах безопасности в настоящее время является серия хеш-функций MD2, MD4, MD5. Все они генерируют дайджесты фиксированной длины в 16 байт. Адаптированным вариантом MD4 является американский стандарт SHA, длина дайджеста в котором составляет 20 байт. Компания IBM поддерживает односторонние функции MDC2 и MDC4, основанные на алгоритме шифрования DES.

ГЛАВА 29 Технологии аутентификации, авторизации и управления доступом

Технологии аутентификации

Как отмечено, аутентификация применительно к вычислительной системе — это доказательство подлинности различных элементов данной системы при их взаимодействии. Пользователь при входе в систему должен предъявить системе доказательства, что он именно тот пользователь, идентификатор которого он вводит. Таким доказательством может служить пароль. Документ, полученный пользователем по электронной почте, должен сопровождаться дополнительной информацией, убеждающей пользователя, что документ не был изменен при передаче и что автором этого документа является именно тот человек, от имени которого это письмо было послано. Здесь доказательством может служить электронная подпись. Устройства, взаимодействующие по сети, должны доказать друг другу, что ни одно из них не подменено злоумышленником с целью ответвления или прослушивания трафика. Для этого в протоколе взаимодействия устройств должна быть предусмотрена процедура взаимной аутентификации. Взаимная аутентификация требуется и для организации безопасного сеанса пользователя и серверного приложения. Аутентификация может проводиться не только по отношению к отдельному пользователю, но и к группе пользователей. Методы аутентификации различаются в зависимости от того, что служит аутентификатором, а также от того, каким образом организован обмен аутентификационными данными между аутентифицируемым и аутентифицирующим элементами системы.

Факторы аутентификации человека

Абсолютно надежная аутентификация человека представляет собой теоретически неразрешимую задачу. Нет такого аутентификатора, который со стопроцентной надежностью доказывал бы аутентичность человека. Пароль можно перехватить, электронный ключ — украсть, отпечаток пальца — подделать, радужную оболочку глаза — подменить качественным изображением. Более того, не существует научного доказательства невозможности совпадения у разных людей отпечатков пальцев или радужных оболочек глаза. Даже совпадение результатов анализа ДНК при современном уровне развития техники не может служить абсолютным доказательством аутентичности человека.

Однако на практике при аутентификации пользователей в вычислительных системах ограничиваются некоторым не стопроцентным, хотя и достаточно высоким уровнем достовер-

ности доказательства аутентичности человека. Аутентификаторы, которые используются при этом, разделяют на три класса:

- «что-то, что знаю» — к этому типу относятся многоразовые и одноразовые пароли, правила преобразования информации;
- «что-то, что имею» — различные миниатюрные устройства, называемые аппаратными аутентификаторами/ключами;
- «что-то, чем являюсь» — различные биометрические показатели аутентифицируемого.

Класс аутентификаторов называют *фактором*. Если в процедуре аутентификации предусматривается предъявление аутентифицируемым нескольких аутентификаторов, относящихся к разным классам, то такую аутентификацию называют многофакторной. Наибольшее распространение в настоящее время получила **двухфакторная аутентификация**, при которой пользователь предъявляет многоразовый пароль («что-то, что знаю») и аппаратный ключ («что-то, что имею»). Следует заметить, что в некоторых случаях термин «многофакторная аутентификация» служит для обозначения процедур **многоступенчатой аутентификации**, построенных на использовании нескольких аутентификаторов, относящихся к одному и тому же классу. Примером такой процедуры является аутентификация владельца банковского счета при его звонке в банк: сначала его просят назвать несколько букв из его пароля, а затем задают несколько вопросов с заранее согласованными и зафиксированными в базе данных аутентифицирующей организации ответами, например, о его памятном географическом пункте, о марке первого автомобиля и т. п.

Аутентификация на основе паролей

Пароль — это используемая при аутентификации сохраняемая в секрете последовательность символов, либо выбранная пользователем, либо сгенерированная программным или аппаратным средством, либо назначенная администратором.

Пароли бывают одноразовыми и многоразовыми. **Многоразовые пароли**, как это следует из их названия, могут использоваться для доказательства аутентичности многократно. В процедурах аутентификации, основанных на **одноразовых паролях**, аутентифицируемый должен каждый раз предъявлять новое значение пароля. Обычно для генерации одноразовых паролей применяются специальные программы или аппаратные устройства (см. далее).

Недостатки многоразовых паролей

Механизмы аутентификации на основе многоразовых паролей, обладая простотой и логической ясностью, традиционно являются самым популярным средством аутентификации. Однако им свойственны и недостатки. Это, во-первых, возможность раскрытия и разгадывания паролей, во-вторых, возможность «подслушивания» пароля при его передаче по сети путем анализа сетевого трафика. В-третьих, обладатели паролей могут стать жертвами социального инжиниринга. Так, например, беглый экс-сотрудник Агентства национальной безопасности США Эдвард Сноуден, работая системным администратором разведывательной базы США на Гавайях, использовал логины и пароли более 20 своих сослуживцев, чтобы получить доступ к секретным файлам. Он получал эти данные, объясняя, что они необходимы ему для работы.

Для снижения уровня угрозы раскрытия паролей администраторы сети, как правило, применяют встроенные программные средства, служащие для формирования *политики назначения и использования паролей*: задание максимального и минимального сроков действия пароля, хранение списка уже использованных паролей, управление поведением системы после нескольких неудачных попыток логического входа и т. п.

Многие пользователи пренебрегают угрозами, которые несут в себе легко угадываемые пароли. Так, червь Mumu, поразивший компьютерные сети в 2003 году, искал свои жертвы, подбирая пароли из очень короткого списка: password, passwd, admin, pass, 123, 1234, 12345, 123456 и пустая строка. Такая на удивление примитивная стратегия дала прекрасные (с точки зрения атакующей стороны) результаты — множество компьютеров было взломано.

В списке наиболее популярных паролей, применяемых пользователями интернета при доступе к веб-серверам, опубликованном в августе 2013 года компанией Google, места в первой десятке занимают имена и даты рождения членов семьи и близких друзей, названия мест рождения, даты свадьбы, клички домашних животных, что-либо связанное с любимой футбольной командой и слово «password». Как видно из приведенного списка, для заинтересованного человека не составит большого труда подобрать эти пароли.

Но даже при выборе менее предсказуемого пароля вы все же рискуете, что он будет разгадан простым перебором всех возможных символов — такой метод часто называют **брутфорс-атакой**[1]. В табл. 29.1 приведены данные, характеризующие стойкость паролей, состоящих из 6 и 8 знаков, сформированных из разных наборов символов. Время определялось для специальной программы подбора паролей, выполняемой на компьютере со средними характеристиками[2]. Обратите внимание, насколько возрастает время подбора пароля при увеличении его длины всего лишь на два знака. Так, при использовании только букв латинского алфавита (строчных и прописных) время подбора пароля из 8 знаков в 3000 раз больше, чем из 6 знаков!

Таблица 29.1. Сравнение стойкости паролей

Множество символов	Количество комбинаций		Время подбора пароля	
	6 знаков	8 знаков	6 знаков	8 знаков
Цифры от 1 до 9	1 миллион комбинаций	100 миллионов комбинаций	Практически мгновенно	10 секунд
26 только прописных или только строчных букв латинского алфавита	309 миллионов комбинаций	200 миллиардов комбинаций	30 секунд	Менее 6 часов (в 720 раз дольше, чем для 6 знаков)
Смесь 52 прописных и строчных букв латинского алфавита	19 миллиардов комбинаций	53 триллиона комбинаций	Полчаса	Два месяца (почти в 3000 раз дольше, чем для 6 знаков)
Прописные и строчные буквы, цифры и все символы (точка, двоеточие и т. п.)	782 миллиарда комбинаций	7,2 квадриллиона комбинаций	22 часа	57 лет (примерно в 22 700 раз дольше, чем для 6 знаков)

[1] Brute-force (англ.) — решать что-либо «в лоб», методом грубой силы.

[2] Данные взяты из статьи http://www.lockdown.co.uk/?pg=combi.

Серьезной проблемой использования многоразовых паролей является их **ручная синхронизация**. В обычной жизни нам требуется не один, а несколько паролей: для входа в сеть предприятия, на котором мы работаем, для доступа к «личному кабинету» провайдера мобильной связи, для доступа к банковскому счету и к другим самым разным интернет-сайтам. Часто во всех этих случаях применяется *один и тот же пароль* (возможно, с небольшими вариациями), потому что у нас нет времени придумывать и, главное, запоминать новый пароль для доступа к новому ресурсу. Выполнив регистрацию на сайте, не заслуживающем доверия, вы сообщаете его владельцам свой пароль, который теперь может быть использован для доступа к другим вашим данным, возможно, имеющим для вас критическое значение.

Слабостью паролей является и процедура реакции аутентифицирующего компьютера на неправильно введенный пароль. На первый взгляд, естественным приемом, направленным на противодействие подбору паролей, кажется *блокирование учетной записи*, с которой было проведено некоторое количество (обычно не более трех) неудачных попыток входа. Однако такой подход дает злоумышленнику прекрасную возможность быстро заблокировать работу предприятия. Действительно, идентификаторы пользователей являются менее защищенной информацией, чем пароли, к тому же они часто легко угадываемы (ADMIN, Guest, IVANOV и т. п.) и их легче подсмотреть, так как они выводятся на экран. Поэтому злоумышленник может легко подобрать имена, выполнить по три неудачные попытки аутентификации для каждой учетной записи, вызвать их блокировку и привести таким образом систему в недоступное состояние. Снятие блокировок с учетных записей может стать серьезной проблемой, если таких записей очень много.

Наряду с паролями существует и другой вариант использования аутентификаторов из класса «что-то, что знаю». Администратор сообщает пользователю заранее безопасным образом некоторое правило, например *правило преобразования* последовательности чисел в другие символы. Во время процедуры аутентификации система выводит на экран случайную последовательность чисел. Пользователь в соответствии с известным только ему и системе правилом преобразует их в другую последовательность символов, которую вводит в качестве пароля. Поскольку система также «знает» правило преобразования, она может проверить правильность введенного пароля. То есть здесь в качестве разделяемого секрета выступает правило преобразования.

Строгая аутентификация в компьютерной сети на основе многоразовых паролей

Как правило, аутентификация пользователей в компьютерных сетях строится на основе централизованной схемы. На одном из серверов сети поддерживается база данных, в которой хранятся *учетные данные* обо всех пользователях сети. Учетные данные содержат наряду с другой информацией идентификаторы и пароли пользователей. Когда пользователь осуществляет логический вход в сеть, он набирает на клавиатуре компьютера свои идентификатор и пароль, которые передаются на сервер. По идентификатору пользователя в централизованной базе данных, хранящейся на сервере, находится соответствующая запись, из нее извлекается пароль и сравнивается с тем, который ввел пользователь. Если они совпадают, то аутентификация считается успешной, пользователь получает легальный статус и те права, которые определены для него системой авторизации.

Однако такая упрощенная схема имеет большой изъян. А именно при передаче пароля с клиентского компьютера на сервер, выполняющий процедуру аутентификации, этот

пароль может быть перехвачен злоумышленником. Поэтому в разных системах аутентификации применяются разные приемы, чтобы избежать передачи пароля по сети в незащищенном виде.

Аутентификация, в процессе которой используются методы шифрования, а аутентификатор не передается по сети, называется **строгой аутентификацией**.

Рассмотрим пример строгой аутентификации пользователей, реализуемой средствами ОС[1]. Пусть аутентификация пользователей сети выполняется на основе их паролей, хранящихся в зашифрованном виде в централизованной базе SAM (Security Accounts Manager). Пароли зашифровываются с помощью односторонней функции шифрования при занесении их в базу данных во время процедуры создания учетной записи для нового пользователя (рис. 29.1). Введем обозначение для этой односторонней функции — ОФШ1. Таким образом, пароль *P* хранится в базе данных SAM в виде дайджеста *d*(*P*), при этом знание дайджеста не позволяет восстановить исходный текст.

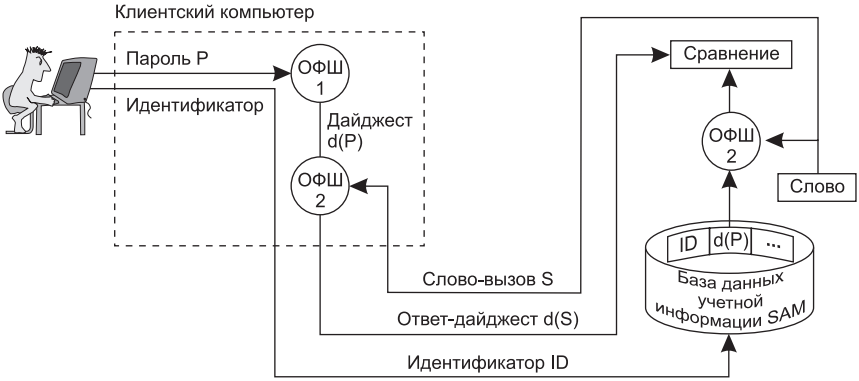


Рис. 29.1. Схема сетевой аутентификации на основе многоразового пароля

При логическом входе пользователь локально вводит в свой компьютер имя-идентификатор (ID) и пароль *P*. Клиентская часть подсистемы аутентификации, получив эти данные, передает запрос по сети на сервер, хранящий базу SAM. В этом запросе в открытом виде содержится идентификатор пользователя, но пароль в сеть ни в каком виде *не передается*.

К паролю на клиентской станции применяется та же односторонняя функция ОФШ1, которая была использована при записи пароля в базу данных SAM, то есть динамически вычисляется дайджест пароля *d*(*P*).

В ответ на поступивший запрос серверная часть службы аутентификации генерирует случайное число *S* случайной длины, называемое **словом-вызовом** (challenge). Это слово передается по сети с сервера на клиентскую станцию пользователя. К слову-вызову на клиентской стороне применяется односторонняя функция шифрования ОФШ2. В отличие от функции ОФШ1, функция ОФШ2 является параметрической и получает в качестве параметра дайджест пароля *d*(*P*). Полученный в результате ответ *d*(*S*) передается по сети на сервер базы SAM.

---

[1] Аутентификация по данной схеме, наряду с другими методами, выполняется в ОС семейства Windows.

Параллельно этому на сервере слово-вызов *S* аналогично шифруется с помощью той же односторонней функции ОФШ2 и дайджеста пароля пользователя *d*(*P*), извлеченного из базы SAM, а затем сравнивается с ответом, переданным клиентской станцией. При совпадении результатов считается, что аутентификация прошла успешно. Таким образом, аутентификация проходит без передачи пароля по каналам связи.

Заметим, что при каждом запросе на аутентификацию генерируется новое слово-вызов, так что перехват ответа *d*(*S*) клиентского компьютера не может быть использован в ходе другой процедуры аутентификации.

Строгая аутентификация в протоколе CHAP

Другим примером строгой аутентификации может служить *аутентификация по квитированию вызова* (Challenge Handshake Authentication Protocol, CHAP), применяемая в протоколе PPP. Протокол PPP предусматривает два режима аутентификации:

- аутентификация по протоколу PAP, когда пароль передается по линии связи в открытом виде;
- аутентификация по протоколу CHAP, при которой пароль по линии связи не передается и, следовательно, обеспечивается более высокий уровень безопасности.

Рассмотрим применение протокола CHAP при аутентификации удаленных пользователей, подключенных к Интернету по коммутируемому каналу. Здесь аутентифицирующей стороной является сервер провайдера, а аутентифицируемой — клиентский компьютер (рис. 29.2). При заключении договора клиент получает от провайдера пароль (пусть, например, это будет слово parol). Этот пароль хранится в базе данных провайдера в виде дайджеста *Z* = *d*(parol), полученного путем применения к паролю односторонней хеш-функции MD5.

В протоколе CHAP предусмотрено четыре типа сообщений: *Success* (успех), *Challenge* (вызов), *Response* (ответ), *Failure* (ошибка).

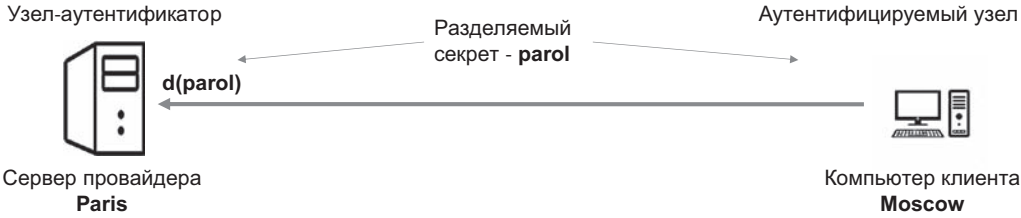


Рис. 27.2. Аутентификация по протоколу CHAP

Аутентификация выполняется в следующей последовательности:

1. Пользователь-клиент активизирует некоторую программу удаленного доступа к серверу провайдера, вводя назначенные ему имя и пароль. Имя (на рисунке это «Moscow») передается по сети провайдеру в составе запроса на соединение, но пароль не передается в сеть ни в каком виде.
2. Сервер провайдера, получив запрос от клиента, генерирует псевдослучайное слово-вызов (пусть это будет слово «goodmorning») и передает его клиенту вместе со значением,

идентифицирующим сообщение в рамках данного сеанса (ID), и собственным именем (здесь — «Paris»). Это сообщение типа *Challenge*: (ID, goodmorning), Paris. Для защиты от перехвата ответа аутентификатор должен использовать разные значения слова-вызова при каждой процедуре аутентификации.

3. Программа клиента, получив этот пакет, извлекает из него слово-вызов, добавляет к нему идентификатор и вычисленный локально дайджест $Z = d(\text{parol})$, а затем вычисляет с помощью все той же функции MD5 дайджест $Y = d\{(\text{ID}, \text{goodmorning}, d(\text{parol}))\}$ от всех этих трех значений. Результат клиент посылает серверу провайдера в пакете *Response*.
4. Сервер провайдера сравнивает полученный по сети дайджест Y с тем значением, которое он получил, локально применив ту же хеш-функцию к набору аналогичных компонентов, хранящихся в его памяти.
5. Если результаты совпадают, то аутентификация считается успешной и аутентификатор посылает партнеру пакет *Success*.

Способ аутентификации, при котором многоразовые пароли пользователей хранятся в базе данных сервера в виде дайджестов, кажется вполне безопасным. Ведь если злоумышленник и сможет получить к ним доступ, то он даже теоретически не сможет восстановить исходное значение паролей по дайджесту. Однако создатель первого червя Роберт Моррис решил эту проблему. Он разработал довольно простую программу, которая генерировала возможные варианты паролей как используя слова из словаря, так и последовательным перебором символов. Для каждого сгенерированного слова вычислялся дайджест, который затем сравнивался с дайджестами из файла паролей. Удивительно, но такая стратегия оказалась весьма эффективной — хакеру удалось завладеть несколькими паролями.

Аутентификация на основе аппаратных аутентификаторов

Алгоритмы аутентификации, основанные на многоразовых паролях, не очень надежны. Пароли можно подсмотреть, разгадать или просто украсть. Более надежными оказываются схемы на основе программных или аппаратных **генераторов одноразовых паролей** (рис. 29.3).

Независимо от того, какую реализацию системы аутентификации на основе одноразовых паролей выбирает пользователь, он, как и в системах аутентификации с применением многоразовых паролей, сообщает системе свой идентификатор, однако вместо того, чтобы вводить каждый раз один и тот же пароль, он указывает последовательность цифр, сообщаемую ему аппаратным или программным ключом. Через определенный небольшой период времени ключ генерирует другую последовательность — новый пароль. Сервер аутентификации проверяет введенную последовательность и разрешает пользователю осуществить логический вход. Сервер аутентификации может представлять собой отдельное устройство, выделенный компьютер или программу, выполняемую на обычном сервере.

Следует иметь в виду, что, как правило, системы аутентификации на основе одноразовых паролей рассчитаны на проверку *удаленных*, а не локальных пользователей.

Рассмотрим схему использования аппаратного генератора одноразовых паролей, в основе которой лежит **синхронизация по времени**. Этот популярный алгоритм аутентификации был разработан компанией Security Dynamics. Идея метода состоит в том, что аппаратный

ключ и аутентифицирующий сервер по одному и тому же алгоритму вычисляют некоторое значение — одноразовый пароль. Алгоритм имеет два параметра:

- *разделяемый секретный ключ*, представляющий собой 64-разрядное число, уникально назначаемое каждому пользователю и хранящееся как в аппаратном ключе, так и в базе данных сервера аутентификации;
- *значение текущего времени*.



Рис. 27.3. Аппаратные ключи, генерирующие одноразовые пароли: *а* — ключ клиентов банка Barklays для доступа к своим счетам; *б* — аппаратный ключ компании SecurID

Если вычисленные значения совпадают, то аутентификация считается успешной.

Итак, пусть удаленный пользователь пытается совершить логический вход в систему с персонального компьютера (рис. 29.4). Аутентифицирующая программа предлагает ему ввести его личный персональный номер (PIN), состоящий из четырех десятичных цифр (на рисунке — 2360), а также одноразовый пароль — шесть цифр случайного числа, отображаемого в тот момент на дисплее аппаратного ключа (на рисунке — 112511). На основе PIN-кода сервер извлекает из базы данных информацию о пользователе, а именно — его секретный ключ. Затем сервер выполняет вычисления по тому же алгоритму, которой заложен в аппаратном ключе, используя в качестве параметров секретный ключ и значение текущего времени, проверяя, совпадает ли сгенерированное число с числом, введенным пользователем. Если они совпадают, то пользователю разрешается логический вход.

Потенциальной проблемой этой схемы является *временная синхронизация* сервера и аппаратного ключа. Вопрос согласования часовых поясов решается просто, но гораздо сложнее обстоит дело с постепенным рассогласованием внутренних часов сервера и аппаратного ключа, тем более что потенциально аппаратный ключ может работать несколько лет. Компания Security Dynamics решает эту проблему двумя способами. Во-первых, при производстве аппаратного ключа измеряется отклонение частоты его таймера от номинала. Далее эта величина учитывается в виде параметра алгоритма сервера. Во-вторых, сервер отслеживает коды, генерируемые конкретным аппаратным ключом, и если таймер данного ключа постоянно спешит или отстает, то сервер динамически подстраивается под него.

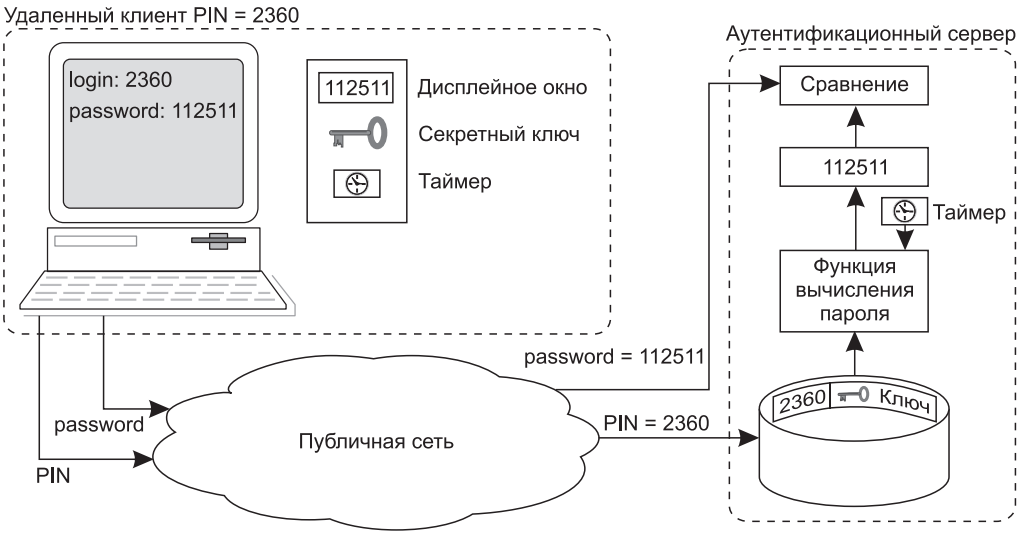


Рис. 29.4. Аутентификация на основе временной синхронизации

Существует, однако, еще одна проблема, связанная со схемой временной синхронизации. Одноразовый пароль, генерируемый аппаратным ключом, действителен в течение некоторого интервала времени (от нескольких десятков секунд до нескольких десятков минут), то есть в течение этого времени одноразовый пароль, в сущности, является многоразовым. Поэтому теоретически возможно, что очень проворный хакер сможет перехватить PIN-код и одноразовый пароль с тем, чтобы также получить доступ в сеть в течение этого интервала.

Схема временной синхронизации *не требует наличия компьютера* на стороне аутентифицируемого — для этих целей можно ограничиться простым терминалом или факсом. Пользователи могут даже вводить свой пароль с телефонной клавиатуры, когда звонят в сеть для получения голосовой почты.

Аутентификация по схеме «запрос-ответ»

Другая схема применения аппаратных ключей, называемая часто **запрос-ответ**, основана на идее, очень сходной с идеей строгой аутентификации, рассмотренной в предыдущем разделе. В том и другом случаях применяется слово-вызов. Когда пользователь пытается осуществить логический вход, аутентификационный сервер передает ему запрос в виде некоторого случайного числа (см. слово-вызов на рис. 29.5). Аппаратный ключ пользователя зашифровывает это случайное число (например, по алгоритму DES) и секретный ключ пользователя. Секретный ключ пользователя хранится в базе данных сервера и в памяти аппаратного ключа. В зашифрованном виде слово-вызов возвращается на сервер. Сервер, в свою очередь, также зашифровывает сгенерированное им самим случайное число с помощью того же алгоритма шифрования и того же секретного ключа пользователя, а затем сравнивает результат с числом, полученным от аппаратного ключа. Как и в методе временной синхронизации, в случае совпадения этих двух чисел пользователю разрешается вход в сеть.

Механизм со словом-вызовом имеет свои ограничения — он обычно требует наличия компьютера на каждом конце соединения, так как аппаратный ключ должен иметь воз-

можность как получать, так и отправлять информацию. Схема «запрос-ответ» уступает схеме временной синхронизации по простоте использования. Для логического входа с помощью схемы временной синхронизации пользователю достаточно набрать 10 цифр. Схемы «запрос-ответ» могут потребовать от пользователя выполнения большего числа ручных действий. В некоторых схемах «запрос-ответ» пользователь должен сам ввести секретный ключ, а затем набрать на клавиатуре компьютера полученное с помощью аппаратного ключа зашифрованное слово-вызов.

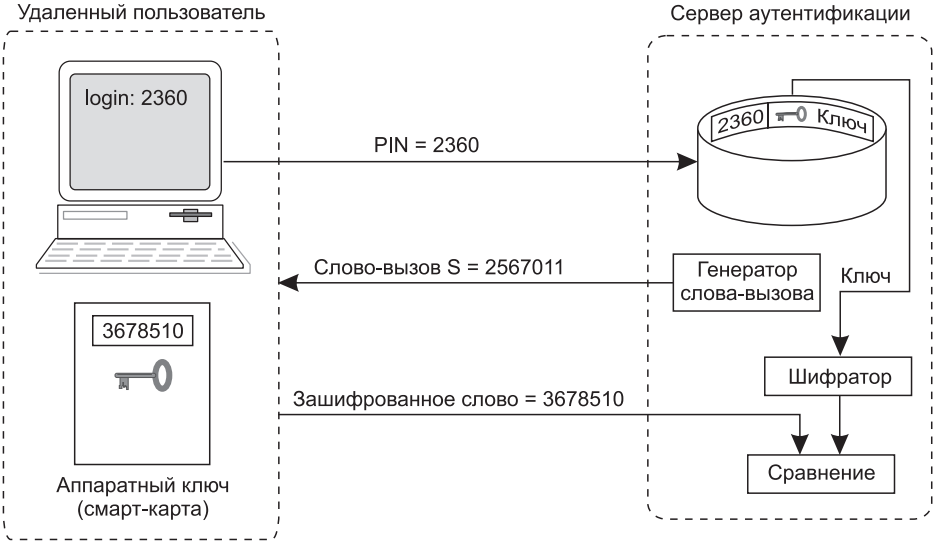


Рис. 29.5. Аутентификация по схеме «запрос-ответ»

Аутентификация информации. Электронная подпись

Аутентификация данных включает:

- подтверждение *целостности* хранящихся и переданных по сети данных и программ, то есть установление факта того, что они не подвергались модификации;
- доказательство *авторства* сообщения (документа, программы), в том числе и для недопущения отказа от авторства;
- доказательство *легальности* приобретения программного обеспечения.

Все эти задачи в той или иной мере могут быть решены посредством электронной подписи.

Согласно терминологии, утвержденной Международной организацией по стандартизации (ISO), под термином «**электронная (цифровая) подпись**» понимаются методы, позволяющие устанавливать подлинность автора сообщения (документа) при возникновении спора относительно авторства. Основная область применения цифровой подписи — финансовые документы, сопровождающие электронные сделки, документы, фиксирующие международные договоренности, и т. п. Подчеркнем, что электронная подпись не ставит задачи обеспечения конфиденциальности сообщений.

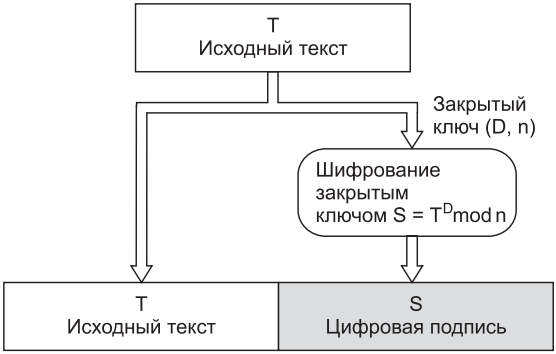


Рис. 29.6. Схема формирования цифровой подписи по алгоритму RSA

Хотя для получения подписи могут использоваться симметричные алгоритмы, более распространенными являются алгоритмы на основе открытого и закрытого ключей. На рис. 29.6 показана схема формирования цифровой подписи по алгоритму RSA. Каждый пользователь сети имеет свой закрытый ключ (D, n), необходимый для формирования подписи, а соответствующий этому секретному ключу открытый ключ (E, n), предназначенный для проверки подписи, известен всем другим пользователям сети. Подписанное сообщение состоит из двух частей: незашифрованной части, в которой содержится исходный текст T, и зашифрованной части, представляющей собой цифровую подпись. Цифровая подпись S вычисляется с помощью закрытого ключа (D, n) по формуле

$$S = T^D \bmod n.$$

Сообщение посылается в виде пары (T, S). Каждый пользователь, имеющий соответствующий открытый ключ (E, n), получив сообщение, отделяет открытую часть T, расшифровывает цифровую подпись S и проверяет равенство

$$T = S^E \bmod n.$$

Если результат расшифровки цифровой подписи совпадает с открытой частью сообщения, то считается, что документ подлинный, не претерпел никаких изменений в процессе передачи, а автором его является именно тот человек, который передал свой открытый ключ получателю.

К недостаткам данного алгоритма можно отнести то, что длина подписи в этом случае равна длине сообщения, что не всегда удобно. Для уменьшения «длины» электронной подписи вместо $S = T^D \bmod n$ используются формула

$$S = (H(T))^D \bmod n.$$

Здесь $H(T)$ — хеш-функция, преобразующая исходное сообщение в короткий дайджест. В этом случае получатель сообщения (T, S) должен сначала применить к открытому тексту T хеш-функцию H и получить дайджест $H(T)$, а затем приступить к расшифровке подписи S открытым ключом. Если расшифрованная подпись совпадает с дайджестом, то авторство сообщения доказано. Использование хеш-функций дает выигрыш не только в объеме сообщения, но и во времени получения электронной подписи.

Если помимо проверки аутентичности документа, обеспечиваемой цифровой подписью, надо обеспечить его конфиденциальность, то после применения к тексту цифровой подписи перед передачей его по каналу связи выполняют совместное шифрование исходного текста и цифровой подписи любым способом шифрования, согласованным отправителем и получателем.

Аутентификация на основе цифровых сертификатов

Аутентификация с применением цифровых сертификатов является альтернативой применению паролей и представляется естественным решением в условиях, когда число пользователей сети (пусть и потенциальных) измеряется миллионами. В таких обстоятельствах процедура предварительной регистрации пользователей, связанная с назначением и хранением их паролей, становится крайне обременительной, опасной, а иногда и просто нереализуемой. При наличии сертификатов сеть, которая дает пользователю доступ к своим ресурсам, не хранит никакой информации о своих пользователях — они ее предоставляют сами в своих запросах в виде сертификатов, удостоверяющих личность пользователей. Сертификаты выдаются специальными уполномоченными сертифицирующими организациями (СО) — **центрами сертификации** (Certificate Authority, CA), или **удостоверяющими центрами**. Поэтому задача хранения секретной информации (закрытых ключей) возлагается на самих пользователей, что делает это решение гораздо более масштабируемым, чем вариант с централизованной базой паролей.

Сертификат представляет собой электронную форму, в которой содержится следующая информация:

- открытый ключ владельца данного сертификата;
- сведения о владельце сертификата, такие, например, как имя, адрес электронной почты, наименование организации, в которой он работает, и т. п.;
- наименование сертифицирующей организации, выдавшей данный сертификат;
- электронная подпись сертифицирующей организации, то есть зашифрованные закрытым ключом этой организации данные, содержащиеся в сертификате.

Использование сертификатов основано на предположении, что сертифицирующих организаций немного и их открытые ключи широкодоступны, например, из публикаций в журналах.

Сертификаты могут быть представлены в трех формах (рис. 29.7):

- *в открытой форме* сертификат содержит всю информацию в незашифрованном виде;
- *в форме из двух частей* — открытой, содержащей всю информацию в незашифрованном виде, и закрытой, представляющей собой ту же информацию, но зашифрованную закрытым ключом сертифицирующей организации;
- *в форме из трех частей* — во-первых, открытой, во-вторых, зашифрованной закрытым ключом сертифицирующей организации, в-третьих, части, представляющей собой первые две части, зашифрованные закрытым ключом владельца.

Когда пользователь хочет подтвердить свою личность, он предъявляет свой сертификат в двух формах: открытой (то есть такой, в которой он получил его в сертифицирующей

организации) и зашифрованной (с применением своего закрытого ключа). Сторона, проводящая аутентификацию, берет из незашифрованного сертификата открытый ключ пользователя и расшифровывает с его помощью зашифрованного сертификата. Совпадение результата с открытым сертификатом подтверждает, что предъявитель действительно является владельцем закрытого ключа, соответствующего указанному открытому.

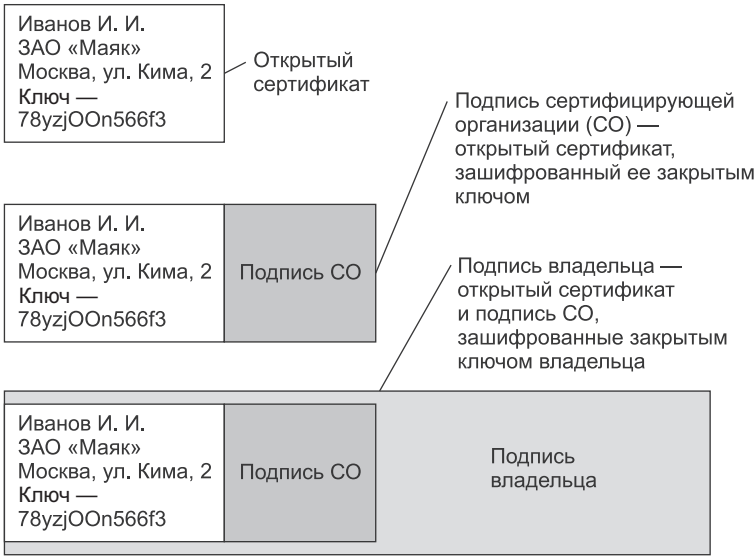


Рис. 29.7. Формы представления цифрового сертификата

Затем с помощью известного открытого ключа указанной в сертификате организации проводится расшифровка подписи этой организации в сертификате. Если в результате получается тот же сертификат с тем же именем пользователя и его открытым ключом, значит, он действительно прошел регистрацию в сертификационном центре, является тем, за кого себя выдает, и указанный в сертификате открытый ключ действительно принадлежит ему.

Сертификаты можно применять не только для аутентификации, но и *для предоставления прав доступа к ресурсам*. Для этого в сертификат могут вводиться дополнительные поля, в которых указывается принадлежность его владельцев к той или иной категории пользователей. Эта категория назначается сертифицирующей организацией в зависимости от условий, на которых выдается сертификат. Например, организация, поставляющая через интернет на коммерческой основе информацию, может выдавать сертификаты определенной категории пользователям, оплатившим годовую подписку на некоторый бюллетень. В этом случае веб-сервер будет предоставлять доступ к страницам бюллетеня только пользователям, предъявившим сертификат данной категории.

Подчеркнем тесную связь открытых ключей с сертификатами. Сертификат является удостоверением не только личности, но и принадлежности открытого ключа.

Цифровой сертификат устанавливает и гарантирует соответствие между открытым ключом и его владельцем .

Это предотвращает угрозу подмены открытого ключа. Если некоторый абонент *А* получает по сети сертификат от абонента *Б*, то он может быть уверен, что открытый ключ, содержащийся в сертификате, гарантированно принадлежит абоненту *Б*, адрес и другие сведения о котором содержатся в этом сертификате. Это значит, что абонент *А* может без опасений использовать открытый ключ абонента *Б* для секретных посланий в адрес последнего.

При наличии сертификатов отпадает необходимость хранить на серверах корпораций списки пользователей с их паролями, вместо этого достаточно иметь на сервере список имен и открытых ключей сертифицирующих организаций. Может также понадобиться некоторый механизм для установления соответствия категорий владельцев сертификатов традиционным группам пользователей, чтобы можно было в неизменном виде задействовать механизмы управления избирательным доступом большинства операционных систем или приложений.

Сертификат является средством аутентификации пользователя при его обращении к сетевым ресурсам, роль аутентифицирующей стороны играют при этом информационные серверы корпоративной сети или интернета. В то же время и сама процедура получения сертификата также включает этап аутентификации, когда аутентификатором выступает сертифицирующая организация. Для получения сертификата клиент должен сообщить сертифицирующей организации свой открытый ключ и те или иные сведения, удостоверяющие его личность. Все эти данные клиент может отправить по электронной почте или принести на съемном носителе лично. Перечень необходимых данных зависит от типа получаемого сертификата. Сертифицирующая организация проверяет доказательства подлинности, помещает свою цифровую подпись в файл, содержащий открытый ключ, и посылает сертификат обратно, подтверждая факт принадлежности данного конкретного ключа конкретному лицу. После этого сертификат может быть встроен в любой запрос на использование информационных ресурсов сети (рис. 29.8).

Практически важным является вопрос о том, кто имеет право выполнять функции сертифицирующей организации. Во-первых, задачу обеспечения своих сотрудников сертификатами может взять на себя само предприятие. В этом случае упрощается процедура первичной аутентификации при выдаче сертификата. Предприятия достаточно осведомлены о своих сотрудниках, чтобы брать на себя задачу подтверждения их личности. Для автоматизации процесса генерации, выдачи и обслуживания сертификатов предприятия могут использовать готовые программные продукты. Например, компания Netscape Communications выпустила сервер сертификатов, который организации могут у себя устанавливать для выпуска своих сертификатов. Во-вторых, эти функции могут выполнять независимые центры по выдаче сертификатов, работающие на коммерческой основе, например сертифицирующий центр компании Verisign. Сертификаты компании Verisign выполнены в соответствии с международным стандартом X.509 и используются во многих продуктах, ориентированных на защиту данных, в том числе в популярном протоколе защищенного канала TSL. Любой желающий может обратиться с запросом на получение сертификата на веб-сервер этой компании.

Механизм получения пользователем сертификата хорошо автоматизируется в сети в модели «клиент — сервер», когда браузер исполняет роль клиента, а в сертифицирующей организации установлен специальный сервер выдачи сертификатов. Браузер генерирует для пользователя пару ключей, оставляет закрытый ключ у себя и передает частично заполненную форму сертификата серверу. Чтобы не подписанный еще сертификат нельзя было подменить при передаче по сети, браузер ставит свою электронную подпись, зашифровывая сертификат выработанным закрытым ключом. Сервер сертификатов подписывает

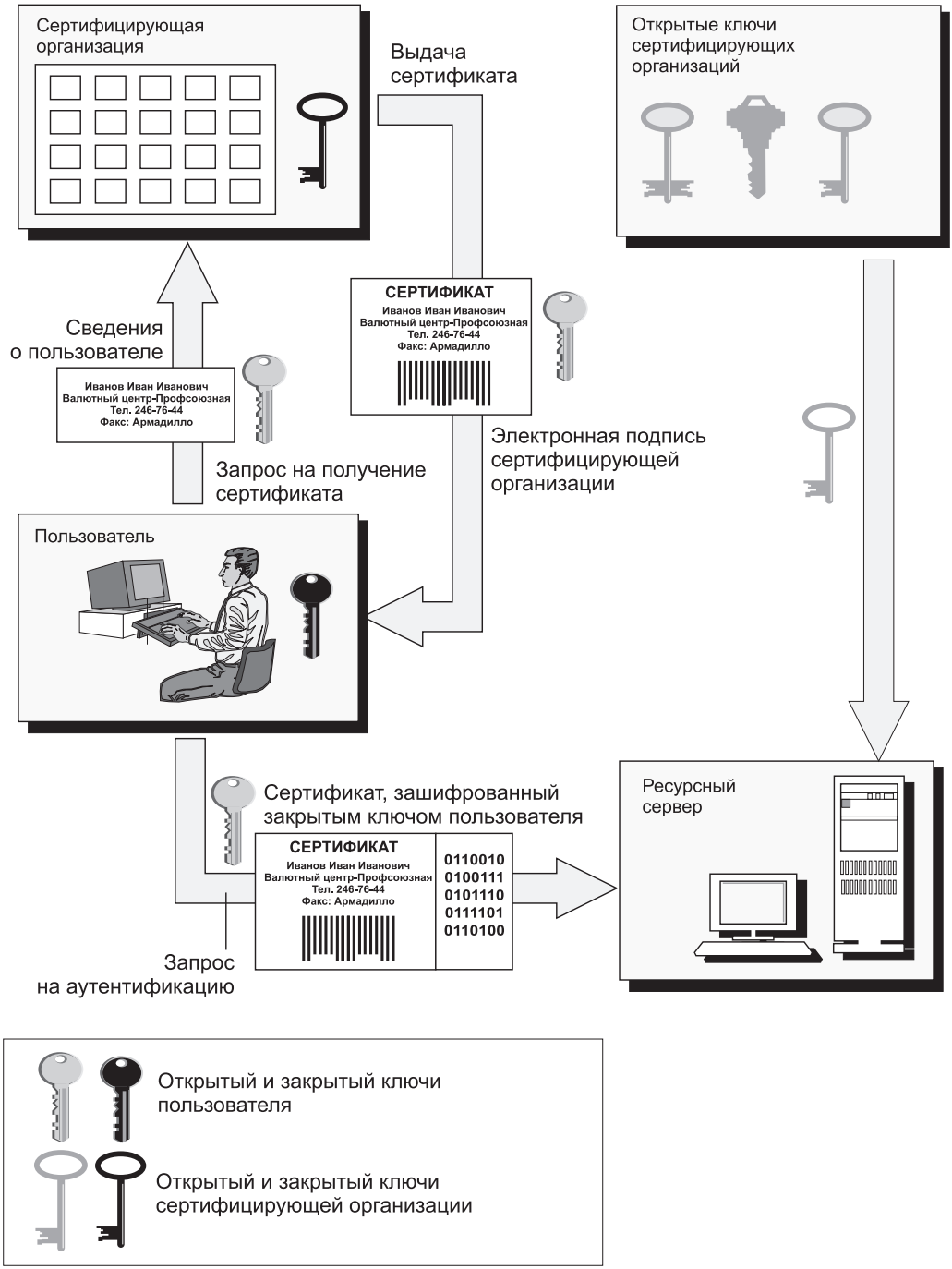


Рис. 29.8. Схема аутентификации пользователей на основе сертификатов

полученный сертификат, фиксирует его в своей базе данных и возвращает его каким-либо способом владельцу. Очевидно, что при этом может выполняться еще и неформальная процедура подтверждения пользователем своей личности и права на получение сертификата, требующая участия оператора сервера сертификатов. Это могут быть доказательства оплаты услуги, доказательства принадлежности к той или иной организации — все случаи жизни предусмотреть и автоматизировать нельзя. После получения сертификата браузер сохраняет его вместе с закрытым ключом и использует при аутентификации на тех серверах, которые поддерживают такой процесс. В настоящее время существует большое количество протоколов и продуктов, применяющих сертификаты. В частности, практически все браузеры и операционные системы реализуют поддержку сертификатов.

Несмотря на активное использование технологии цифровых сертификатов во многих системах безопасности, эта технология еще не решила целый ряд серьезных проблем. Это прежде всего поддержание базы данных о выпущенных сертификатах. Сертификат выдается не навсегда, а на некоторый вполне определенный срок. По истечении срока годности сертификат должен либо обновляться, либо аннулироваться. Кроме того, необходимо предусмотреть возможность досрочного прекращения полномочий сертификата. Все заинтересованные участники информационного процесса должны быть вовремя оповещены о том, что некоторый сертификат уже недействителен. Для этого сертифицирующая организация должна оперативно поддерживать список отозванных сертификатов.

Имеется также ряд проблем, связанных с тем, что сертифицирующие организации существуют не в единственном числе. Все они выпускают сертификаты, но даже если эти сертификаты соответствуют единому стандарту (сейчас это, как правило, стандарт X.509), то все равно остаются нерешенными многие вопросы. Все ли сертифицирующие центры заслуживают доверия? Каким образом можно проверить полномочия того или иного сертифицирующего центра? Можно ли создать иерархию сертифицирующих центров, когда сертифицирующий центр, стоящий выше, мог бы сертифицировать центры, расположенные в иерархии ниже? Как организовать совместное использование сертификатов, выпущенных разными сертифицирующими организациями?

Для решения этих и многих других проблем, возникающих в системах, использующих технологии шифрования с открытыми ключами, оказывается необходимым комплекс программных средств и методик, называемый *инфраструктурой с открытыми ключами* (Public Key Infrastructure, PKI).

Рассмотренная схема аутентификации включает три основных элемента — это пользователи, цифровые сертификаты и центры сертификации. Чтобы данная схема работала надежно и эффективно, в нее должны быть включены дополнительные элементы, которые в совокупности с основными и образуют PKI.

В число дополнительных элементов может входить, например, регистрационный центр (Registration Authority), который служит посредником между пользователем, запросившим сертификат, и центром сертификации. Пользователь обычно обращается к регистрационному центру с помощью веб-интерфейса и сообщает данные о себе. Регистрационный центр проверяет эту информации и в случае ее подлинности передает данные о пользователе, подписанные собственным закрытым ключом, центру сертификации. Регистрационный центр может обслуживать несколько центров сертификации. При отсутствии регистрационного центра его функции выполняет центр сертификации.

Другим типом дополнительных элементов PKI являются разнообразные хранилища сертификатов, содержащие информацию о действующих, отозванных и истекших сертификатах.

Аутентификация программных кодов

Электронная подпись и сертификаты могут применяться для доказательства аутентичности (подлинности) программ. Пользователю важно быть уверенным, что программа, которую он загрузил с какого-либо сервера интернета, действительно содержит коды, разработанные определенной компанией. Компания Microsoft предложила для этих целей технологию **аутентикода** (authenticode).

Организация, желающая подтвердить свое авторство на программу, должна встроить в распространяемый код так называемый **подписывающий блок** — аутентикод (рис. 29.9). Этот блок состоит из двух частей. Первая часть — это сертификат организации-разработчика данной программы, полученный обычным образом от какого-либо сертифицирующего центра. Вторую часть образует зашифрованный дайджест, полученный в результате применения хеш-функции к распространяемому коду. Шифрование дайджеста выполняется с помощью закрытого ключа организации.

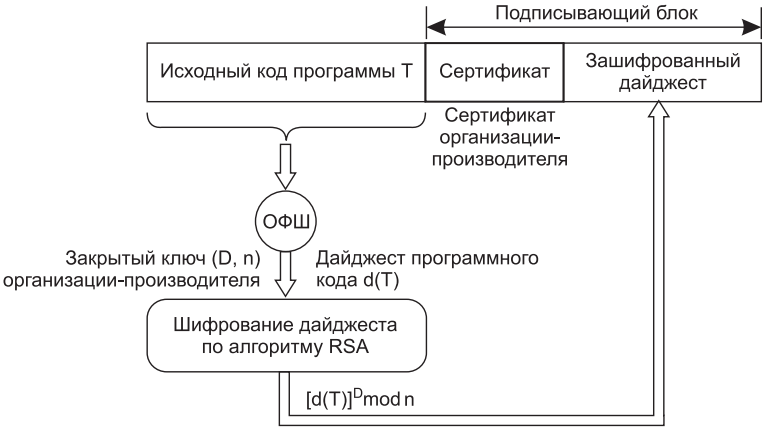


Рис. 29.9. Схема получения аутентикода

Компания-разработчик может потребовать от пользователя программы доказательство легальности ее приобретения — допустим, запросить регистрационный номер программы (Product ID или Serial Number), называемый также *лицензионным ключом активации*. Обычно этот номер пишется на отдельном бланке, прилагаемом к поставляемой программе, наносится на упаковку или высылается по электронной почте при покупке программы через интернет.

Другим способом доказательства легальности приобретения и законности использования программных продуктов являются миниатюрные электронные устройства — *электронные замки*, подобные уже рассмотренным нами аппаратным аутентификаторам. Эти устройства поставляются вместе с защищаемыми от нелегального использования программами. Перед запуском программы электронный замок должен быть подключен к компьютеру, например, через USB-порт. Инициирующий блок программы обращается к данному устройству

с запросом и, получив правильный ответ, начинает работать. Если же ожидаемый ответ не поступает, то выполнение программы блокируется. Таким образом, электронный замок действует как специфический аутентификатор пользователя, доказывающий то, что он является законным владельцем программы.

(S) *Биометрическая аутентификация*

Аутентификация пользователей ОС

Существует две принципиально отличные схемы аутентификации, реализуемые операционными системами и специальными сетевыми службами. В одной из них, которую мы будем называть **локальной системой аутентификации**, ОС работает в пределах одного компьютера: она задействует базу аутентификационных данных пользователей, причем результаты аутентификации могут применяться только для доступа к ресурсам этого компьютера.

По другой схеме работает так называемая **система аутентификации домена**: она базируется на центральной базе аутентификационных данных пользователей группы компьютеров (*домена аутентификации*), хранящейся на одном из серверов сети, и результаты аутентификации служат для доступа к ресурсам данного домена.

Локальная система аутентификации ОС работает при *логическом входе пользователя* как с терминала *компьютера*, так и *через* сеть. Первый вариант называют **интерактивным** логическим входом, второй — **удаленным** (сетевым или неинтерактивным). Понятно, что при удаленном логическом входе риски безопасности выше, так как аутентификационные данные передаются через сеть — корпоративную или интернет — и их легче перехватить. Перехват данных аутентификации представляет собой угрозу даже в случае строгой аутентификации, когда пароль не передается в открытом виде по сети или же не передается вовсе — при наличии большого массива аутентификационных данных, то есть данных перехватов большого количества процедур входа одного и того же пользователя, пароль может быть вычислен по имеющимся результатам его ввода.

Хотя локальные системы аутентификации ОС поддерживают все распространенные методы аутентификации — на основе многоразовых и одноразовых паролей (аппаратных и программных), биометрических данных и цифровых сертификатов, — основным методом аутентификации пользователей является метод на базе *многоразового пароля*. Практически все универсальные ОС (MS Windows, Unix/Linux и MAC OS X) предлагают этот метод по умолчанию.

Одноразовые пароли, обеспечивающие более надежную аутентификацию, чем многоразовые, чаще применяются при удаленном логическом входе через соединения VPN с шифрованием информации, где передача аутентификационной информации идет через интернет и, следовательно, риск ее перехвата и взлома особенно велик. Одноразовые пароли могут сочетаться с многоразовыми при двухфакторной аутентификации.

Аутентификация на основе *сертификатов* применяется чаще всего для удаленно работающих пользователей, которые предъявляют сертификаты, выданные сервером сертификации, организации, к которой принадлежит пользователь.

Аутентификация на основе *биометрических данных* штатными средствами универсальных ОС обычно не поддерживается, так как их повышенная надежность нужна только в особо

защищенных системах. Кроме того, для поддержки биометрической аутентификации требуется приобрести и установить соответствующее специальное программное обеспечение и специальные устройства.

Необходимо отличать процедуру аутентификации *пользователя ОС* от процедуры аутентификации *пользователя серверной части* некоторого приложения. Многие серверные приложения имеют собственную систему аутентификации пользователей, никак не связанную с системой аутентификации ОС, под управлением которой они работают. Независимость системы аутентификации сервера приложений имеет как положительные, так и отрицательные стороны. Преимуществом здесь является разграничение по умолчанию пользователей ОС, которым потенциально может понадобиться доступ к любому ресурсу компьютера, и пользователей некоторого сервиса, которым нужен доступ только к ресурсам, относящимся к данному сервису, например, только к файлам, хранящимся в корневом каталоге FTP-сервера. К недостаткам же можно отнести низкую защищенность протокола аутентификации некоторых приложений, а также необходимость запоминания двух различных имен и паролей для одного и того же пользователя, ошибки администраторов ОС и сервисов из-за дублирования учетных записей и т. п.

Технологии управления доступом и авторизации

После того как пользователь, пройдя аутентификацию, доказал свою легальность, ему предоставляется некоторый набор прав по отношению к защищаемым системой ресурсам.

> Наделение легальных пользователей правами доступа к ресурсам называется **авторизацией**. Процедура приведения авторизации в действие называется **управлением доступом** (access control).

Если, например, субъект пытается использовать ресурс с запрещенным для него типом доступа, то механизм управления доступом должен отклонить эту попытку и, возможно, уведомить систему об этом инциденте с целью генерации сигнала тревоги.

Формы представления ограничений доступа

При решении задачи управления доступом необходимо руководствоваться *принципом минимальных привилегий*. В соответствии с ним каждому субъекту в системе должен быть назначен минимально возможный набор прав, достаточный для решения именно тех задач, на которые он уполномочен. Применение этого принципа ограничивает те возможные потери, которые могут быть нанесены в результате неумышленных ошибок или неавторизованных действий.

Ограничение доступа может задаваться в форме **правил**. На основании правила система управления доступом в любой момент времени *динамически* решает вопрос о предоставлении или непредоставлении доступа. Правило может строиться с учетом различных факторов, в том числе длительности сеанса связи (ограничение доступа по времени использования ресурса), возраста человека (ограничение для детей на доступ к некоторым сайтам), времени суток (разрешение на использование ресурсов и сервисов интернета только в рабочие часы).

Популярной мерой ограничения доступа в Интернет является *капча* (capcher) — субъекту, обратившемуся с запросом к ресурсу, предлагается ввести символы, выведенные на экран в таком искаженном виде, в котором их сможет распознать только человек, — таким образом исключается доступ к ресурсам искусственных субъектов (программных систем).

Для ограничения доступа используются также **контентно**- и **контекстно-зависимые правила**. Например, в компании может быть принято правило, в соответствии с которым некоторым категориям пользователей запрещается доступ к документам, содержащим те или иные ключевые слова или фразы: «для ограниченного использования», «секретно», слова, обозначающие кодовое название проекта, и др. Ограничения могут быть наложены и на доступ к ресурсам, содержащим текст на иностранном языке. Это примеры контентно-зависимых правил. В контекстно-зависимых правилах принимаются во внимание некоторые факторы, характеризующие текущее состояние среды и/или предысторию (контекст) запроса. Простейшим правилом такого рода является отказ в доступе пользователю, который сделал подряд три безуспешные попытки аутентификации.

Эффективным средством ограничения доступа является **конфигурирование пользовательского интерфейса**. Таким путем пользователь может быть лишен возможности не только обращаться к тем или иным каталогам и файлам, но и возможности видеть на своем экране часть структуры файловой системы, доступ к которой ему запрещен. Администратор может настроить систему меню пользовательского интерфейса так, что некоторые пункты этих меню не будут выводиться на экран, что исключит принципиальную возможность запуска пользователем части функций.

Матрица прав доступа является универсальной и наиболее гранулированной (то есть тонко дифференцированной) формой представления политики контроля доступа, она директивно, «в лоб» описывает для каждого пользователя набор конкретных операций, которые ему разрешается выполнять по отношению к каждому объекту (рис. 29.10).

Субъекты / Объекты	User 1	User 2	User 3
File 1	Читать и записывать	Читать и записывать	Читать
File 2	Читать и записывать	Нет доступа	Читать
File 3	Записывать	Нет доступа	Читать

Рис. 29.10. Матрица прав доступа

Матричный способ описания прав доступа теоретически дает возможность отразить все многообразие отношений субъектов и объектов системы для всех возможных сочетаний {субъект, объект, назначенные права}. Однако этот универсальный способ представления, как правило, очень сложно реализовать на практике из-за громоздкости матрицы, учитывая огромное число элементов — как субъектов, так и объектов — в вычислительной системе.

Особенностью матрицы прав доступа является не только ее большая размерность, но и наличие большого числа *нулевых* элементов. Такой вид матриц в математике называют

разряженными. Нулевое значение здесь говорит о том, что для данного сочетания (субъект, объект) права доступа не определены, а именно такие сочетания составляют большинство в реальных системах. Свойство разреженности матрицы может быть использовано для более компактного представления правил доступа.

С каждым объектом можно связать **список управления доступом** (Access Control List, ACL), в котором указаны только те субъекты (пользователи), которые имеют разрешение на доступ к данному объекту (файлу). Ясно, что количество субъектов в данном списке будет значительно меньше общего числа субъектов системы. Такие списки должны быть созданы для всех ресурсов. Способ описания прав доступа набором списков столь же универсальный и гибкий, как матрица, но вместе с тем имеет и более компактный вид, поскольку не включает пустые элементы матрицы (рис. 29.11, *а*).

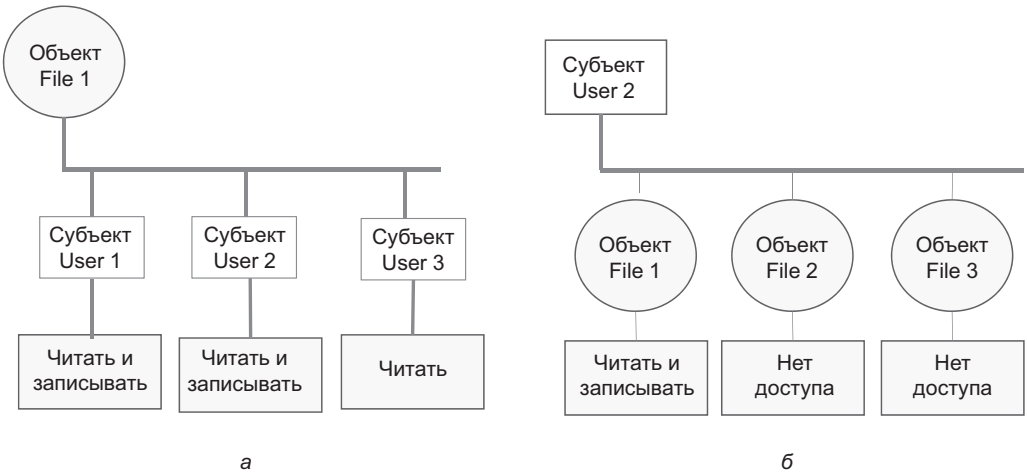


Рис. 29.11. *а* — список управления доступом к объекту; *б* — список разрешений пользователя User 2

Права доступа могут быть определены как по отношению к ресурсам, так и по отношению к пользователям. В последнем случае его называют **списком разрешений** (capability). На рис. 29.11, *б* показан список разрешений, которые имеет пользователь User 2 по отношению к ресурсам File 1, File 2 и File 3.

Очевидно, что совокупность списков управления доступом ко всем ресурсам системы несет ту же самую информацию, что и совокупность списков разрешений для всех пользователей, так как и те и другие являются разными проекциями одной и той же матрицы. В одних реализациях систем управления доступом (например, в большинстве операционных систем) применяются ограничения, заданные для объекта (ACL), а в других (например, в некоторых расширениях системы Kerberos, включающих авторизацию) — ограничения для субъекта (списки разрешений).

Другим способом «сжатия» матрицы является определение прав доступа *для групп субъектов* по отношению к *группам объектов*. Такое представление возможно, когда многие элементы матрицы имеют одинаковое значение, что соответствует ситуации в реальной системе, когда некоторая группа пользователей имеет одинаковые права.

Это дает возможность компактно описать права доступа с помощью матрицы меньшей размерности.

В некоторых случаях, если существует простое *правило* определения прав доступа, хранение матрицы вообще не требуется, поскольку значения ее элементов могут вычисляться системой управления доступом *динамически*. Например, пусть все объекты и субъекты системы изначально снабжены метками из одного и того же множества. Кроме того, предположим для простоты изложения, что для всех объектов определен только один вид операции доступа. Допустим также существование следующего правила: доступ к объекту разрешен, если метки субъекта и объекта совпадают, и не разрешен, если не совпадают. Имея такое правило, нет смысла заранее создавать и хранить матрицу — проще вычислять соответствующий элемент при каждой попытке доступа.

Ранее мы рассматривали различные подходы к хранению и представлению информации о правах доступа, не придавая значения тому, каким именно образом они были назначены. Однако способ назначения прав — *авторизация* — существенно влияет на способ управления доступом.

Существует два основных подхода к авторизации:

- для авторизации выделяется особый *полномочный орган* (authority), который принимает все решения о наделении пользователей правами относительно всех объектов;
- функции принятия решений по авторизации *делегируются* некоторым субъектам.

Каждый из этих двух подходов управления доступом может быть реализован множеством различных способов, отражающих разные методы задания и приведения в исполнение ограничений, однако большинство реализуемых на практике способов может быть отнесено к одной из следующих категорий:

- **дискреционный метод доступа** (Discretionary Access Control[1], **DAC**), называемый также избирательным или произвольным;
- **мандатный метод доступа** (Mandatory Access Control[2], **MAC**), называемый также принудительным;
- **ролевой доступ** (Role-based Access Control, **RBAC**), называемый также недискреционным методом доступа (nondiscretionary access control).

Помимо этих методов в чистом виде, система управления доступом может базироваться и на их комбинации.

Дискреционный метод управления доступом

Одно из первых систематических изложений принципов DAC было предпринято в 1987 году в документе NCSC-TG-003-87, «Руководство по дискретному управлению доступом». В то время модель DAC была самой распространенной схемой управления доступом и, кстати, таковой она остается и по сегодняшний день — большинство универсальных ОС реализуют дискреционную модель. В документе NCSC дается следующее определение метода DAC:

---

[1] Discretionary — действующий по своему усмотрению.

[2] Mandatory — обязательный, принудительный.

Дискреционный метод представляет собой средство ограничения доступа к объектам, базирующееся на уникальных идентификаторах субъекта и/или групп, к которым этот субъект относится. Управление доступом в методе DAC является дискреционным, или произвольным, — в том смысле, что субъект, обладающий некоторыми разрешениями на доступ к объектам, может по своему усмотрению передать часть своих полномочий (иногда прямо, а иногда — опосредованно) другим субъектам.

Отсюда следуют две главные особенности дискреционного метода:

- Права доступа в методе DAC описываются в виде *списков ACL*, которые дают возможность гибкого и гранулированного определения набора разрешенных операций для каждого отдельного пользователя по отношению к каждому отдельному ресурсу, причем и пользователи, и ресурсы задаются уникальными идентификаторами.
- В методе DAC право назначать права на доступ к объектам *делегируются* отдельным пользователям — владельцам объектов. То есть им разрешается действовать по своему усмотрению и назначать другим пользователям права на доступ к тем объектам, владельцами которых они являются.

Таким образом, процедура авторизации является распределенной между множеством пользователей-владельцев. Владельцами считаются пользователи, создавшие объект, или пользователи, которые были назначены владельцами другими уполномоченными на то пользователями или системными процессами. Владелец имеет полный контроль над созданным им объектом и несет всю полноту ответственности за управление доступом к нему. Вместе с тем он может назначать права доступа к своим объектам, руководствуясь некоторым правилом, принятым на предприятии.

Основным достоинством метода DAC является его гибкость, обусловленная свободой пользователей наделять правами или аннулировать права других пользователей на доступ к своим ресурсам, а также возможностями тонкой настройки набора разрешенных операций. Однако это достоинство имеет свою оборотную сторону.

Как и всякая распределенная система, система управления доступом по методу DAC страдает от *невозможности гарантированно проводить общую политику*, осуществлять надежный контроль действий пользователей. Любая политика безопасности, принятая на предприятии, может быть нарушена в результате ошибочных или вредительских действий пользователей.

Другой недостаток дискреционного метода связан с тем, что здесь *права на доступ определяются по отношению к объекту*, а не его содержимому. Это означает, что любой пользователь (точнее, его процесс), имеющий доступ к файлу согласно некоторому списку ACL1, может скопировать его содержимое в другой файл, характеризуемый другим списком ACL2. Это показывает, что системы с контролем доступа по методу DAC не могут применяться там, где требуется очень высокий уровень защиты информации.

(S) *Слабость метода DAC*

Мандатный метод управления доступом

Мандатный доступ позволяет реализовать системы, отвечающие самым строгим требованиям безопасности, — как правило, они используются в правительственных и военных учреждениях или в других организациях, для которых чрезвычайно важен высокий уровень защиты данных.

К основным чертам мандатного метода управления доступом можно отнести следующие:

- авторизацию и управление доступом осуществляет *центральный полномочный орган*, отвечающий за безопасность (обычно в роли такого органа выступает ОС);
- решение о предоставлении права доступа принимается ОС динамически на основе простого *правила*, которое разрабатывается уполномоченными на то лицами на основе политики безопасности.

Простота правил достигается тем, что и субъекты, и объекты разбиваются на небольшое число групп. Каждой группе объектов присваивается **уровень (гриф) секретности**, а группам субъектов — **уровни допуска** к объектам того или иного уровня секретности. В разных системах могут быть приняты разные правила, но все они базируются на сравнении уровня секретности объекта и уровня допуска субъекта. Например, правило может быть следующим: субъекту разрешается доступ к объекту, если уровень его допуска равен или выше уровня секретности объекта. На рис. 29.12 это правило представлено в виде матрицы.

Уровень допуска субъектов / Уровень секретности объектов	Уровень от «совершенно секретно» и ниже	Уровень от «секретно» и ниже	Уровень данных для служебного пользования
Совершенно секретно	Доступ разрешен	0	0
Секретно	Доступ разрешен	Доступ разрешен	0
Данные для служебного пользования	Доступ разрешен	Доступ разрешен	Доступ разрешен

Рис. 29.12. Правило мандатного доступа, представленное в виде матрицы

Пользователи должны принимать решение системы как данность, они лишены возможности управлять доступом к своим ресурсам или передавать свои права другим пользователям. В отличие от систем DAC, мандатный доступ *имеет централизованный характер и позволяет жестко проводить принятую политику безопасности.*

Элементы, описывающие уровни секретности объектов или уровни допуска субъектов, называют **метками безопасности** (security labels). Мандатный метод управления доступом предусматривает назначение меток безопасности всем без исключения субъектам и объектам системы с тем, чтобы в дальнейшем они использовались системой для принятия решений о допуске.

В большинстве случаев для адекватного отражения политики безопасности невозможно сформулировать правило, основанное на учете только уровней секретности и допусков. К одному и тому же уровню секретности могут быть отнесены самые разные материалы, а в соответствии с принципом минимальных привилегий пользователь должен получать доступ только к той информации, которую ему необходимо знать. Чтобы сделать возможным более специфическое задание прав доступа, в метки безопасности объекта и субъекта добавляется информация о конкретном виде данных, к которому относится данный объект или к которому разрешен доступ данному субъекту соответственно.

Таким образом, каждая метка безопасности состоит из двух частей (рис. 29.13):

- часть, отражающая уровень секретности/допуска, называется *классификацией*;
- часть, характеризующая специфику информации, называется *категорией*.



Рис. 29.13. Структура метки безопасности объекта/субъекта

Категория относит данные к определенному виду информации. Например, разные категории могут быть присвоены материалам, относящимся к разным проектам, разным административным подразделениям, разным профессиональным группам. Одному и тому же объекту/субъекту может быть присвоено несколько категорий. Так, отчет о завершении этапа некоторой антитеррористической операции может быть отнесен не только к категории материалов, касающихся данной операции, но и дополнительно к категории материалов подразделения, занимающегося этой работой. Объекты одной категории могут быть классифицированы по-разному, например, одна часть отнесена к более высокому уровню секретности, а другая часть — к более низкому.

Правило, определяющее право доступа, строится на анализе обеих частей меток безопасности объекта и субъекта. Доступ разрешается, если выполняются следующие два условия:

- классификация субъекта равна или выше классификации объекта;
- по меньшей мере одна из категорий объекта, к которому пытается получить доступ субъект, совпадает хотя бы с одной из категорий данного субъекта.

Рисунок 29.14 иллюстрирует соотношение между классификацией и категорией. Здесь цвет кружков служит для обозначения разных категорий объектов. На рисунке показано три уровня классификации: «совершенно секретно», «секретно» и «для служебного пользования». Объекты одной категории могут принадлежать разным уровням классификации. В метке безопасности субъекта указана классификация «секретно» и перечислены две категории, к которым ему разрешен доступ. Стрелками показаны три попытки доступа. Попытка обращения к уровню «совершенно секретно» была заблокирована системой из-за недостаточно высокого уровня допуска субъекта. Обращение к объекту уровня «секретно» было разрешено, так как классификация субъекта равна классификации объекта, а категория объекта совпала с одной из категорий, указанных в метке безопасности субъекта. Попытка доступа к объекту уровня «для служебного пользования» была пресечена, хотя субъект и имеет более высокий уровень допуска («секретно»). В данном случае ограничением служит категория объекта, которая не совпадает ни с одной из категорий субъекта.

Мандатный доступ, как уже отмечалось, является более безопасным, чем дискреционный, но для его эффективной реализации требуется большой объем подготовительной работы,

а после запуска системы необходимо поддерживать в актуальном состоянии метки безопасности существующих объектов, а также назначать метки новым ресурсам и пользователям.

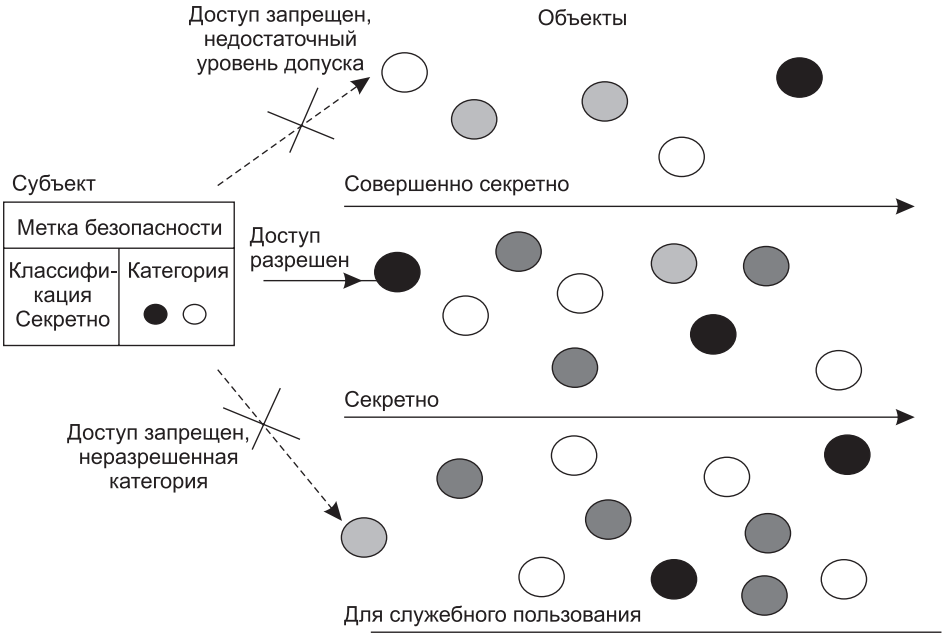


Рис. 29.14. Правило мандатного доступа

Ролевое управление доступом

Ролевой метод управления доступом более приближен к реальной жизни, чем дискреционный и мандатный методы. Как видно из названия, основным его свойством является использование «ролей».

Понятие «роль» в данном контексте ближе всего к понятию «должность» или «круг должностных обязанностей». Поскольку одну и ту же должность может занимать несколько людей, то и одна и та же роль может быть приписана разным пользователям. Роли устанавливаются для целей авторизации. Набор ролей должен некоторым образом (не однозначно) соответствовать перечню различных должностей, существующих на предприятии, к которому эта система относится. Ролевая система доступа лучше всего работает в организациях, в которых существует четкое распределение должностных обязанностей.

Разрешения приписываются ролям, а не отдельным пользователям или группам пользователей (рис. 29.15). Затем те или иные роли приписываются пользователю. Например, в системе управления доступом, развернутой в банке, всем юристам приписана роль «юрист», трейдерам — роль «трейдер», менеджерам — роль «менеджер» и т. д. Процесс определения ролей должен включать тщательный анализ того, как функционирует организация, какой набор функций должен выполнять работник, имеющий ту или иную должность. Каждой из ролей назначаются права доступа, необходимые и достаточные пользователям для выполнения служебных обязанностей, обусловленных приписыванием к данной роли.

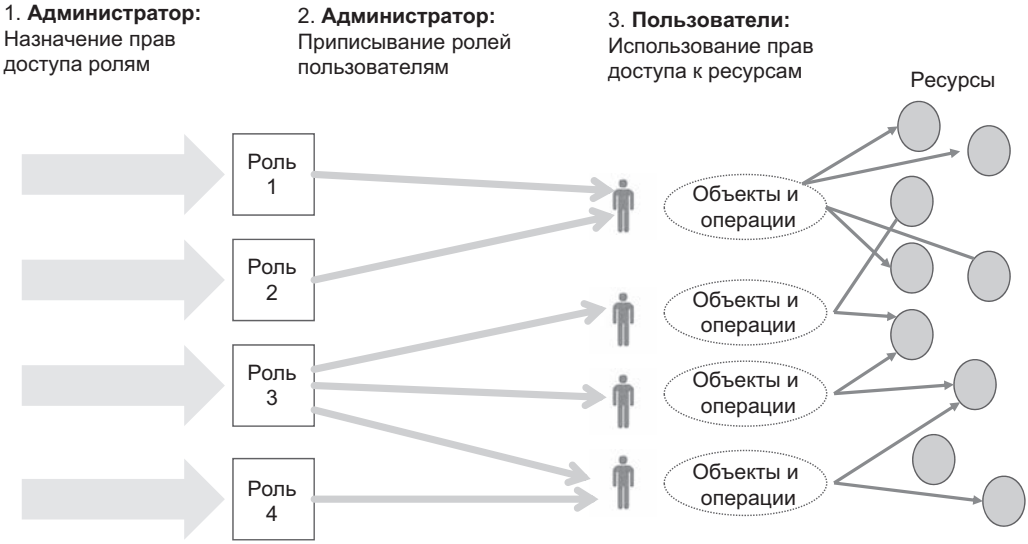


Рис. 29.15. Схема авторизации в системах управления доступом на основе ролей

Каждому пользователю может быть приписано *несколько* ролей (с некоторыми ограничениями, о которых рассказано далее). Во время сеанса работы пользователя все роли, которые ему назначены, становятся активными и пользователь получает права доступа, являющиеся результатом объединения прав доступа всех этих групп.

Все пользователи, играющие одну и ту же роль, имеют идентичные права. Изменение производственной ситуации: расширение бизнеса, внедрение новых технологий, продвижение сотрудника по служебной лестнице, перевод сотрудника в другое подразделение и др. — все это может вызвать аннулирование одной роли пользователя и приписывание ему другой роли. Такой подход упрощает администрирование прав доступа: вместо необходимого в дискреционном и мандатном методах отслеживания и обновления прав каждого отдельного пользователя, в ролевом методе достаточно изменить роль или заменить одну роль другой. Таким образом, к особенностям ролевого управления доступом можно отнести следующее:

- Ролевой метод управления доступом *сочетает в себе черты мандатного и дискреционного методов.*
- Ролевую систему управления доступом *легче администрировать* и контролировать, чем дискреционную. В дискреционной системе права назначаются пользователю «мелкими порциями»: запись в определенный файл, чтение другого файла, запуск некоторой программы и т. п. Такой способ позволяет с ювелирной точностью создавать и индивидуально настраивать комплекс прав доступа пользователя, однако он является очень трудоемким, вследствие чего возрастает возможность ошибок. В ролевой системе права доступа выдаются в виде «глыбы» — интегрированного набора разрешений, рассчитанных на возможность выполнения некоторых относительно сложных операций: заполнение кредитного документа, генерация отчетов и др.
- Ролевой доступ является *централизованным* методом — так же, как и в мандатном методе, пользователь лишен возможности управлять назначением прав. Назначение

пользователю роли можно считать некоторым аналогом приписывания уровня допуска пользователю мандатной системы. Однако ролевой доступ является более гибким, чем мандатный, а по возможностям настройки прав доступа ролевой доступ ближе к дискреционному.

- Ролевой метод доступа нельзя отнести к хорошо масштабируемым. Он эффективно работает в пределах единой системы или приложения, таких, например, как FreeBSD, СУБД Oracle, MS Active Directory, но на больших предприятиях, численность персонала которых составляет тысячи сотрудников, поддержание множества ролей становится сложной и запутанной задачей. Занимая промежуточное положение между мандатным и дискреционным методами, ролевое управление доступом уступает им обоим в масштабируемости. В мандатном методе централизованный характер принятия решений (который не способствует масштабируемости) компенсируется простотой выполняемого алгоритма назначения прав. В дискреционном же методе, напротив, сложность механизма наделения правами компенсируется распределенным характером процедуры принятия решений.

(S) *Ролевое управление доступом. Иерархия ролей, отношение наследования*

(S) *Формальные модели безопасного управления доступом*

Управление доступом в операционных системах

Что касается систем управления доступом в *универсальных ОС*, то там доминирует *дискреционная модель* управления доступом; согласно этой модели владелец ресурса (пользователь, который его создал или которому передано владение) самостоятельно определяет, кто имеет доступ к этому ресурсу и какие операции с ним он может выполнять. Практически все популярные сегодня семейства универсальных ОС – Unix/Linux/CentOS/Ubuntu, Mac OS X, MS Windows – опираются на дискреционную модель доступа как основную.

Мандатная модель – это особенность *специализированных ОС*, рассчитанных на применение в среде с повышенными требованиями к безопасности. Тем не менее существует набор модулей ядра Linux под названием SELinux (Security Enhanced Linux), который реализует многие свойства мандатной модели в среде Linux.

Ролевая модель применяется в универсальных ОС частично в виде механизма встроенных групп с предопределенными правами, сосуществуя с моделью дискреционного доступа для индивидуальных пользователей и групп.

(S) *Аутентификация и авторизация в операционных системах*

Централизованные системы аутентификации и авторизации

Традиционный способ аутентификации с помощью многоразовых паролей отлично подходит для случая, когда пользователь все время работает с единственным компьютером, обращаясь только к его ресурсам и ресурсам интернета, не требующим аутентификации. Такому пользователю нужно запоминать и периодически менять только один пароль. Однако более типичным является случай, когда пользователю приходится работать на

разных географически рассредоточенных компьютерах — рабочем стационарном, компьютере домашнем стационарном компьютере, личном планшете, гостевом компьютере предприятия-партнера — и при этом получать доступ к различным серверам: серверам своего предприятия, серверам предприятия-партнера, к защищенным веб-сайтам интернета и др.

В том случае, когда каждый компьютер и каждый сервер требуют отдельной аутентификации с помощью многоразового пароля, пользователю приходится помнить и обновлять довольно много паролей, и с этой задачей многие пользователи справляются не очень успешно. Согласно исследованию, проведенному Network Applications Consortium, около 70 % звонков пользователей в службу ИТ-поддержки связано с просьбой восстановления забытого пароля, а в среднем пользователь крупной корпоративной сети тратит на процедуры логического входа 44 часа в год. Неудивительно, что большие усилия затачиваются на разработку процедур так называемого единого логического входа.

Целью **единого логического входа** (Single Sign On, SSO) является создание такого порядка аутентификации, при котором пользователь выполняет вход в сеть только один раз, доказывая свою аутентичность с помощью любого способа аутентификации, а затем результат этой аутентификации прозрачным для пользователя способом применяется каждый раз, когда ему нужно доказывать свою аутентичность какому-либо другому серверу или приложению.

В настоящее время не существует системы аутентификации, реализующей концепцию единого логического входа, которая бы работала со всеми типами операционных систем, приложений и при этом учитывала бы разнообразные отношения между организациями, к которым принадлежат пользователи и информационные ресурсы. Однако имеются системы, позволяющие организовать единый логический вход для однородной в каком-то отношении информационной системы, например, для сети, использующей только одну определенную ОС или один определенный протокол аутентификации, либо для группы организаций, доверяющих друг другу при аутентификации своих пользователей. Так, например, свойство однородности операционных систем является условием применимости системы единого входа на основе справочной службы Microsoft Active Directory.

Схема, иллюстрирующая идею систем единого логического входа, представлена на рис. 29.16.

В этой схеме имеются три элемента:

- *Пользовотель*, который располагает некоторой информацией, достаточной для его аутентификации. Это может быть информация любого типа из упомянутых ранее — многоразовый пароль, одноразовый пароль, цифровой сертификат, биометрические данные и т. п. На рисунке в качестве примера показан вариант аутентификации на основе многоразового пароля,здесъ ID — идентификатор, PW — *многоразовый паролъ*.
- *Провайдер идентичности* (Identity Provider) — это система, которая может аутентифицировать пользователя на основе базы данных учетных записей пользователей. Этот элемент может иметь и другие названия, например *сервер аутентификации*.
- *Провайдер сервисов* (Service Provider), называемый также ресррсньдлг серверов, — это система, предоставляющая сервисы пользователям. Такими сервисами могутбыть файловый сервис, почтовый сервис, веб-сервис, сервис базданных и т. п. Предполагается, что сервис предоставляется только аутентифицированным пользователям.

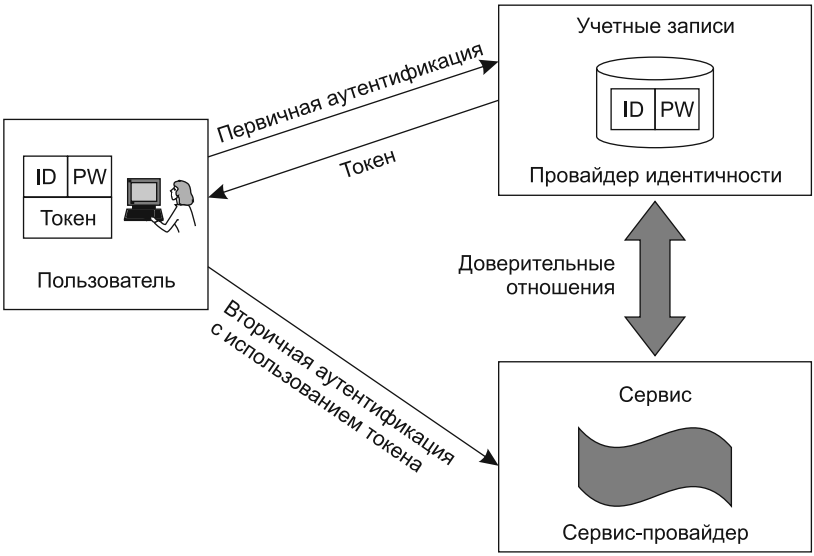


Рис. 29.16. Схема единого логического входа

Особенность схемы — в том, что база учетных данных имеется только у провайдера идентичности, а провайдер сервисов доверяет результатам аутентификации пользователей, выполненной провайдером идентичности. Говорят, что в таком случае существуют *доверительные отношения* (trust relationships) между провайдером идентичности и провайдером сервисов.

Пользователь выполняет логический вход в сеть, обращаясь к провайдеру идентичности. Если пользователь смог подтвердить свою аутентичность, то провайдер идентичности предоставляет пользователю некоторую информационную структуру — *токен доступа*, который пользователь хранит в своей базе данных. При необходимости получения доступа к некоторому сервису пользователь предъявляет токен доступа ресурсному серверу. Токен доступа защищен криптографически таким образом, что ресурсный сервер имеет возможность убедиться, что токен был выдан пользователю сервером аутентификации, которому ресурсный сервер доверяет аутентифицировать пользователей. Говорят, что в этом случае происходит *вторичная аутентификация* пользователя, но для самого пользователя она прозрачна, так как предъявлением токена доступа занимается программное обеспечение его компьютера.

Токен доступа обычно имеет ограниченное время действия, например сутки, поэтому пользователь должен его возобновлять, повторяя процедуру с сервером аутентификации.

Примером системы аутентификации, реализующей концепцию единого логического входа, является протокол и основанная на нем сетевая служба Kerberos.

(S) *Классификация систем единого логического входа*

(S) *Система Kerberos*

ГЛАВА 30 Технологии безопасности на основе анализа трафика

Фильтрация

Под **фильтрацией трафика** понимается обработка IP-пакетов маршрутизаторами и файерволами, приводящая к отбрасыванию некоторых пакетов или изменению их маршрута. Фильтрация трафика позволяет либо предотвратить атаку на сеть, заранее блокируя доступ к ней для некоторых внешних сетей и хостов, либо, если источник атаки не был предварительно заблокирован, остановить ее.

Условия фильтрации бывают самыми разными, поэтому не всегда удается найти простой признак, по которому одни пакеты нужно пропускать, а другие — отбрасывать. К тому же такое условие почти всегда является компромиссом между предотвращением атаки и поддержанием должной функциональности защищаемого узла — чем больше потенциальных атак мы предотвращаем, тем больше урезаем функции узла TCP/IP, связанные с его обычной работой. Поэтому фильтрацию можно рассматривать как инструмент, который может быть как полезным, так и опасным, поэтому им надо уметь пользоваться. Еще одним аргументом в пользу тщательного рассмотрения условий фильтрации перед ее применением является потенциальное замедление продвижения трафика при его фильтрации, так как проверка условий фильтрации — это дополнительное действие, и маршрутизатор или файервол может не справиться с такой дополнительной работой вовремя.

Виды фильтрации

Выборочная передача кадров/пакетов маршрутизатором осуществляется на основе стандартных и дополнительных правил, называемых также **фильтрами**.

Стандартные правила фильтрации присущи не только маршрутизаторам, но и другим коммуникационным устройствам — концентраторам и коммутаторам. Эти правила определяют способ функционирования устройств. Так, стандартное (функциональное) правило фильтрации для *концентратора* заключается в том, что кадр, поступивший на любой его интерфейс, независимо от адреса назначения кадра *повторяется* на всех остальных его интерфейсах. *Коммутатор* же функционирует в соответствии с правилом, когда кадр, имеющий некоторый адрес назначения, повторяется только на том интерфейсе, к которому подключена подсеть, имеющая в своем составе узел с данным адресом. Что касается стандартных правил фильтрации для *маршрутизатора*, то они состоят в том, что пакет, поступивший на входной интерфейс, перемещается на тот или иной интерфейс (или отбрасывается) *на основе адресной таблицы* маршрутизатора, в которой учитываются параметры маршрутов и пакетов.

Дополнительные правила фильтрации, или **пользовательские фильтры**, задаются сетевыми администраторами, исходя из политики безопасности либо с целью изменения стандартных маршрутов. Дополнительные правила фильтрации маршрутизаторов[1] могут учитывать:

- IP-адреса источника и приемника;
- MAC-адреса источника и приемника;
- идентификаторы интерфейсов, с которых поступают пакеты;
- типы протоколов, сообщения которых несут IP-пакеты (TCP, UDP, ICMP, OSPF);
- номера портов TCP/UDP (то есть типы протоколов прикладного уровня).

При наличии пользовательского фильтра маршрутизатор сначала сравнивает описываемые этим фильтром условия с признаками пакета и при положительной проверке выполняет над пакетом ряд нестандартных действий. Например, пакет может быть отброшен; направлен следующему маршрутизатору, отличающемуся от того, который указан в таблице маршрутизации; помечен как вероятный кандидат на отбрасывание при возникновении перегрузки. Одним из таких действий может быть и обычная передача пакета в соответствии с записями таблицы маршрутизации.

Фильтрация может преследовать *разные цели*, в том числе логическую структуризацию сети, нестандартную маршрутизацию, защиту сети от вредительского трафика.

Логическая структуризация, то есть разделение компьютерной сети на подсети и сегменты, самым непосредственным образом влияет на ее эффективность. Инструментом логической структуризации является фильтрация *пользовательского трафика*, проходящего через интерфейсы сетевых устройств на основе стандартных правил.

Нестандартная маршрутизация реализуется за счет фильтрации *маршрутных объявлений*. Протоколы маршрутизации, обмениваясь маршрутными объявлениями, создают таблицы маршрутизации, на основе которых любой узел составной сети может связываться с любым другим узлом. Благодаря этому принципу дейтаграммных сетей каждый пользователь Интернета может получить доступ к любому публичному сайту (напомним, в сетях, основанных на технике виртуальных каналов, действует другое правило: взаимодействие произвольных узлов невозможно без предварительной процедуры установления между ними виртуального канала). Однако такая всеобщая достижимость узлов в IP-сетях не всегда отражает потребности их владельцев, поэтому многие маршрутизаторы поддерживают фильтрацию объявлений протоколов маршрутизации, что позволяет дифференцированно управлять достижимостью узлов. Подчеркнем, фильтрация трафика в целях безопасности является важным средством **защиты от атак**. Функцию фильтрации поддерживают файерволы разного типа, в том числе файерволы на базе маршрутизаторов.

Правила фильтрации маршрутизаторов Cisco

Рассмотрим примеры пользовательских фильтров, написанных на командном языке маршрутизаторов Cisco. Эти фильтры, называемые **списками доступа** (access list)[2], явля-

---

[1] О фильтрации трафика коммутаторами локальных сетей см. главу 11.

[2] Напомним, что термин «список доступа» употребляется и при описании механизмов контроля доступа в ОС, но в ином смысле.

ются очень распространенным средством ограничения пользовательского трафика в IP-маршрутизаторах. Существует два типа списков доступа Cisco:

- **стандартный список доступа** (Standard), позволяющий задавать условия фильтрации, учитывающие только IP-адрес источника;
- **расширенный список доступа** (Extended), позволяющий использовать в условиях фильтрации IP-адреса источника и приемника, порты TCP и UDP источника и приемника, а также типы сообщений некоторых других протоколов, например ICMP.

Как стандартный, так и расширенный список доступа может состоять из нескольких условий, каждое из которых записывается в виде отдельной строки. Условия применяются к пакету в том порядке, в котором они перечисляются в списке доступа до первого совпадения (оставшиеся условия не проверяются).

Стандартный список доступа имеет следующий формат:

```
access-list номер_списка_доступа { deny | permit } {адрес_источника [ метасимволы_
источника ] |any }
```

Служебные слова стандартного списка доступа:

- `access-list` — служебное слово, с которого начинается каждая запись;
- `deny` — запрет прохождения пакета, если условие выполняется;
- `permit` — разрешение прохождения пакета, если условие выполняется;
- `any` — служебное слово, которое говорит о том, что условие должно быть применено к любому значению адреса источника.

Числовые параметры стандартного списка доступа:

- *номер_списка_доступа* — всем условиям одного и того же списка доступа присваивается один и тот же номер из диапазона 1–99;
- *адрес_источника* — IP-адрес источника;
- *метасимволы_источника* используются аналогично маске, которая накладывается на поля IP-адреса источника поступившего пакета и сравнивается с параметром *адрес_источника*.

Пример стандартного списка доступа:

```
access-list 1 deny 192.78.46.0 0.0.0.255
```

Здесь:

- 1 — номер списка доступа;
- `deny` — пакет, который удовлетворяет условию данного списка доступа, должен быть отброшен;
- 192.78.46.0 — адрес источника;
- 0.0.0.255 — метасимволы источника.

Этот фильтр запрещает передачу пакетов, у которых в старших трех байтах адреса источника имеется значение 192.78.46.0.

Список доступа может включать более одного условия. В этом случае он состоит из нескольких строк с ключевым словом access-list с одним и тем же номером. Так, если мы хотим разрешить прохождение через маршрутизатор пакетов хоста 192.78.46.12, запрещая

передачу пакетов, отправляемых любым другим хостом подсети 192.78.46.0/24, то список доступа будет выглядеть следующим образом:

```
access-list 1 permit 192.78.46.12 0.0.0.0
access-list 1 deny 192.78.46.0 0.0.0.255
```

Расширенный список доступа имеет следующий формат:

```
access-list номер_списка_доступа { deny | permit } ключевое_слово_протокола
{адрес_источника метасимволы_источника [ операция порт_источника] | any }
{ адрес_приемника метасимволы_приемника [ операция порт_приемника ] | any }
```

Параметры:

- *номер_списка_доступа* — номер списка доступа из диапазона 100–199;
- *ключевое_слово_протокола* — `ip`, `tcp`, `udp` или `icmp`;
- *операция*: `eq`, `lt`, `gt` (позволяет задать порт, диапазон портов UDP/TCP или тип пакета ICMP).

Расширенный список дает возможность фильтровать пакеты определенных приложений на основе известных портов TCP/UDP их серверной части. Рассмотрим несколько примеров. Пример 1:

```
access-list 105 permit tcp any host 210.135.17.101 eq 21
```

Эта запись разрешает прием запросов от любого хоста, направленных FTP-серверу (TCP-порт 21) с адресом 210.135.17.101 (используется дополнительное служебное слово `host` вместо маски 0.0.0.0). Пример 2:

```
access-list 101 deny ICMP any 192.78.46.0 0.0.0.255 eq 8
```

Эта запись запрещает передачу эхо-запросов (ping-запросов) от любого хоста к хостам подсети 192.78.46.0/24. Пример 3:

```
access-list 105 permit tcp any eq 80 any gt 1023 established
```

Эта запись разрешает клиентам веб-службы (они всегда имеют порт TCP > 1023) получать ответы от любых веб-серверов (порт 80), с которыми у них уже установлено TCP-соединение (служебное слово `established` оговаривает это, маршрутизатор проверяет данный факт по наличию признака ACK в пакете).

Список доступа можно применять к любому интерфейсу маршрутизатора и в любом направлении: если список применяется с ключевым словом `in`, то он действует на входящие в интерфейс пакеты. В этом случае говорят, что выполняется **входная фильтрация** (ingress filtering). Например, написанный нами список доступа 1 можно применить к некоторому интерфейсу для обработки входящего трафика, используя следующую команду:

```
access-list 1 in
```

Если же применить список доступа с ключевым словом `out`, то он будет воздействовать на пакеты, исходящие из интерфейса, и в этом случае будет выполняться **выходная фильтрация** (egress filtering).

Для обеспечения подотчетности необходимо *протоколирование событий*, связанных с фильтрацией пакетов. Маршрутизаторы Cisco могут помещать сообщения об обработке

пакетов, удовлетворяющих условию некоторой записи списка доступа, в системный журнал маршрутизатора syslog. По умолчанию такая опция для каждой записи списка доступа неактивна — это сделано для уменьшения нагрузки на маршрутизатор. Для активизации протоколирования необходимо добавить к записи ключевое слово `log`, например:

```
access-list 102 permit TCP any 21 any log
```

В заключение отметим, что приведенный здесь пример языка для списков доступа маршрутизаторов Cisco является хотя и фирменной, но достаточно типичной реализацией, хорошо иллюстрирующей возможности применения маршрутизаторов как файерволов. Отсутствие фильтрации с запоминанием состояния связано со стремлением не создавать слишком большую нагрузку на маршрутизатор и «не отвлекать» его от основных обязанностей. Это ограничение является главным отличием маршрутизаторов от программных и программно-аппаратных файерволов.

(S) *Фильтрация маршрутных объявлений*

Файерволы

Функциональное назначение файервола

Файервол (межсетевой экран, или **брандмауэр)** — это комплекс программно-аппаратных средств, осуществляющий информационную защиту одной части компьютерной сети от другой путем анализа и фильтрации проходящего между ними трафика.

Файерволы осуществляют экранирование защищаемого объекта и формируют его внешнее представление. В сетевой среде файерволы часто являются первым и самым мощным рубежом обороны.

ПРИМЕЧАНИЕ

Исходным значением термина «файервол» (от англ. firewall) является элемент конструкции дома, а именно — стена, сделанная из огнеупорного материала и препятствующая распространению огня между частями дома (обычно принадлежащими разным собственникам). Термин «брандмауэр» (от нем. brandmauer) много лет назад пришел в русский язык из немецкого. Изначально он обозначал перегородку в поезде, отделяющую область топки паровоза от пассажирского отделения. Интересно, что немецкие специалисты в области безопасности для обозначения межсетевого экрана используют англоязычный термин firewall. В русском языке для термина «файервол» используются и другие транслитерации: файрволл, файрвол, фаервол.

Файервол (рис. 30.1) защищает внутреннюю сеть (например, локальную сеть предприятия или, как вырожденный случай, отдельный компьютер пользователя) от угроз, исходящих из внешней сети (в общем случае — из интернета). Файервол может также защищать одну внутреннюю сеть предприятия от другой, если в соответствии с принципом минимума полномочий пользователям этих сетей не требуется полный взаимный доступ к ресурсам друг друга.

Для эффективного выполнения файерволом его главной функции — анализа и фильтрации трафика — необходимо, чтобы через него проходил *весь* трафик, которым обмениваются узлы защищаемой части сети с узлами интернета. Если сеть связана с внешними сетями несколькими линиями связи, то каждая такая линия должна быть защищена файерволом.

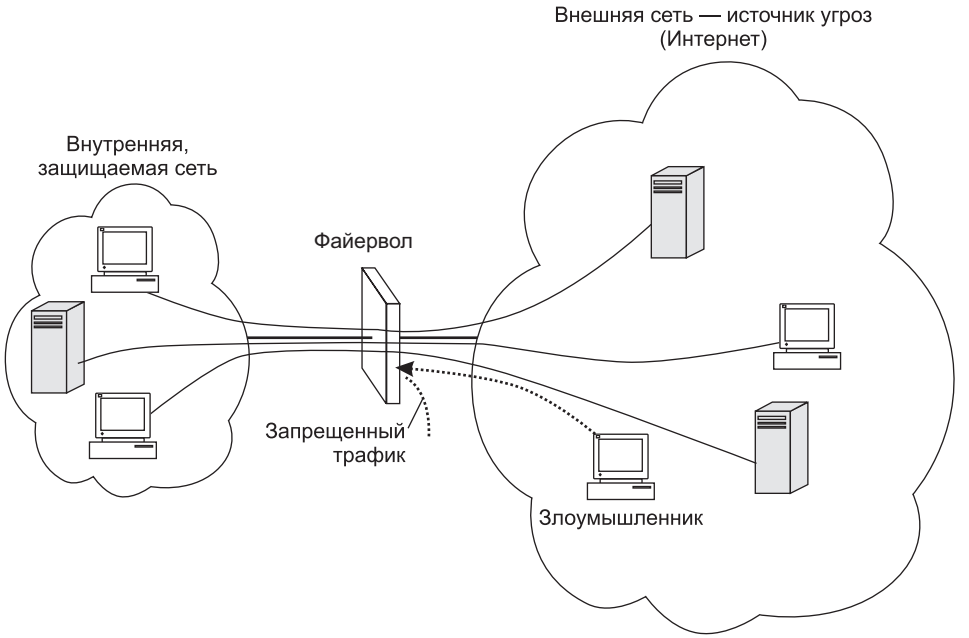


Рис. 30.1. Файервол защищает внутреннюю сеть от угроз, исходящих из внешней сети

Основными функциями файервола являются:

- фильтрация трафика в целях защиты внутренних ресурсов сети;
- аудит — файервол должен фиксировать все события, связанные с обнаружением и блокировкой подозрительных пакетов.

Наряду с этими двумя базовыми функциями на файервол могут быть возложены и другие *вспомогательные функции* защиты, в частности:

- антивирусная защита;
- шифрование трафика;
- логическое посредничество между внутренними клиентами и внешними серверами (функция прокси-сервера);
- трансляция сетевых адресов (NAT);
- фильтрация сообщений по содержимому, включая типы передаваемых файлов, имена DNS и ключевые слова;
- предупреждение и обнаружение вторжений и сетевых атак;
- функции VPN.

Как можно заметить, большинство из перечисленных функций часто реализуются в виде отдельных продуктов или в составе систем защиты других типов. Так, функции пакетной фильтрации встроены практически во все маршрутизаторы; задача обнаружения вирусов решается множеством разнообразных программ; шифрование трафика — неотъемлемый элемент технологий защищенных каналов и т. д. и т. п. Прокси-серверы часто поставляются в виде приложений, более того, они сами иногда интегрируют в себе функции, свойствен-

ные межсетевым экранам, такие, например, как фильтрация по содержимому (контенту) или трансляция сетевых адресов.

Отсюда возникают сложности при определении понятия «файервол». Например, довольно распространено мнение, что файервол — это пограничное устройство, выполняющее фильтрацию пакетов (то есть маршрутизатор), а прокси-сервер — это совершенно отличный от файервола инструмент защиты. Другие настаивают, что прокси-сервер является непременным и неотъемлемым атрибутом любого файервола, третьи — что файерволом может быть названо только такое программное (аппаратное) устройство, которое способно отслеживать состояние потока пакетов в рамках соединения. Мы же в этой книге будем придерживаться следующей точки зрения:

Файервол — это программно-аппаратный комплекс, выполняющий разнообразные функции по защите внутренней сети, набор которых может меняться в зависимости от типа, модели и конкретной конфигурации файервола, при этом минимальный набор функций должен включать фильтрацию трафика для предотвращения сетевых атак и аудит событий, связанных с фильтрацией.

Пример-аналогия

Функционально сетевой экран можно сравнить с системой безопасности современного аэропорта. Аналогии здесь достаточно очевидные (рис. 30.2) — самолет соответствует защищаемой внутренней сети, а внешняя сеть, из которой приходит потенциально опасный трафик, — внешнему миру, откуда прибывают будущие пассажиры самолета, готовящегося к полету, при этом не все они приезжают с чистыми и ясными намерениями.



Рис. 30.2. Зона контроля аэропорта как аналогия файервола

В потоке пассажиров, постоянно входящих в здание аэропорта, могут встречаться различные злоумышленники. Наиболее зловещие — террористы — пытаются пронести на борт взрывчатку (в сетевом мире — пакеты, несущие во внутреннюю сеть вирусы, способные «взорвать» серверы и компьютеры пользователей) или оружие для захвата самолета в воздухе (атака по захвату управления удаленным компьютером). Контрабандисты несут с собой незадекларированные ценности (запрещенный контент), а некоторые личности пытаются попасть в самолет по поддельным документам (несанкционированный доступ к внутренним ресурсам сети).

Чтобы отфильтровать трафик пассажиров, система безопасности аэропорта пропускает всех пассажиров и их багаж через единственно возможный путь — зону контроля. Так же поступают при защите сети, направляя весь входящий трафик через файервол. В зоне контроля аэропорта применяются разнообразные средства проверки пассажиров и их багажа. Аутентификация происходит путем сличения паспортов с компьютерной базой данных, а лиц пассажиров — с фотографиями в паспортах. Сюда же можно отнести просвечивание сумок и чемоданов, проход пассажиров через металлодетекторы. Между злоумышленниками и службой безопасности постоянно происходит состязание в коварстве, с одной стороны, и находчивости — с другой. Например, использование террористами флаконов для маскировки жидких компонентов бомбы лишило пассажиров возможности брать с собой в салон шампуни и другие жидкости в больших объемах.

Файерволы тоже пытаются использовать все возможные средства и методы для противостояния разнообразным угрозам. С помощью паролей и цифровых сертификатов они проверяют аутентичность внешних узлов, пытающихся установить соединения с внутренними; отслеживают логику обмена пакетами для того, чтобы отразить атаки, основанные на искажении этой логики; «просвечивают» содержимое электронных писем и загружаемых документов, пытаясь блокировать запрещенный контент; сканируют загружаемые программы, проверяя их на наличие известных вирусов. Так же, как и в зоне контроля аэропорта, здесь постоянно идет соревнование между хакерами, все время изобретающими новые методы атак, и разработчиками файерволов, старающихся эти атаки обнаружить и пресечь.

Типы файерволов

Файерволы можно классифицировать по самым разным критериям. Остановимся в этой связи на способе реализации, способе фильтрации и уровне модели OSI.

Способ реализации файервола так же многовариантен, как и его функциональность. В качестве аппаратной составляющей сетевого экрана может выступать маршрутизатор или комбинация маршрутизаторов, компьютер или комбинация компьютеров, комбинация маршрутизаторов и компьютеров, наконец, это может быть и какое-то специализированное устройство. Таким же разнообразием отличается и программная составляющая сетевого экрана, имеющая гибкую структуру и включающая в себя различные модули, функции которых могут широко варьироваться. В самом общем виде по способу реализации различают программный, аппаратный и программно-аппаратный файерволы:

- **программный файервол** реализован как программная система, работающая под управлением универсальной ОС, такой как Microsoft Windows, Linux или Mac OS (возможно, имеющая версии для нескольких универсальных ОС);
- **аппаратный файервол** реализован как набор дополнительных функций маршрутизатора, относящихся к фильтрации (реже — Ethernet-коммутатора);
- **программно-аппаратный файервол** представляет собой программную систему, работающую на базе специализированной платформы. Обычно в качестве такой специализированной платформы выступает аппаратный сервер, сертифицированный для работы

с программным обеспечением файервола, плюс установленная на нем универсальная ОС с набором специфических настроек, обеспечивающих максимальный уровень безопасности.

В зависимости от *способа фильтрации* различаются файерволы без запоминания состояния и файерволы с запоминанием состояния:

- **файерволы без запоминания состояния** (stateless) выполняют фильтрацию на основе статических правил, при этом не отслеживаются состояния соединений (сеансов);
- **файерволы с запоминанием состояния** (stateful) принимают решения динамически, с учетом текущего состояния сеанса и его предыстории.

Файерволы с запоминанием состояния для каждого сеанса, который удовлетворяет некоторым условиям, создают динамическую структуру данных в специальной таблице состояний файервола. После прихода очередного пакета контролируемого сеанса состояние сеанса корректируется и принимается решение о выполнении заданного действия с пакетом — пропускать или отбрасывать. Отслеживание для протоколов состояний сеансов требует больших объемов ресурсов, именно поэтому файерволы с запоминанием состояния создаются на программно-аппаратных платформах, имеющих большую оперативную память для хранения таблицы состояний сеансов и быстродействующие процессоры для обработки в реальном времени поступающих пакетов. Если же ресурсов такого файервола оказывается недостаточно, то вместо пользы он может принести вред — например, в ситуации, когда внутренние серверы оказываются недоступными не из-за атак на них, а из-за заторов трафика на интерфейсах файервола.

Теперь, когда мы обсудили такую особенность файерволов, как способность работать в режимах с запоминанием и без запоминания состояния, можно сказать, что именно отсутствие у маршрутизаторов режима фильтрации с запоминанием состояния является *наиболее существенным их отличием* от программных и программно-аппаратных файерволов. Это ограничение связано со стремлением не создавать слишком большую нагрузку на маршрутизатор, чтобы «не отвлекать» его от выполнения основных обязанностей. В то же время существуют и исключения из этого правила, например, модели Juniper SRX являются, с одной стороны, файерволами, поддерживающими фильтрацию с запоминанием состояния и работающими на всех уровнях, включая прикладной, а с другой — они поддерживают все функции «нормального» маршрутизатора, а не только их усеченный набор, как это часто происходит с программно-аппаратными файерволами.

Одной из наиболее важных характеристик файервола является *уровень протокола модели OSI*, на котором он работает. По этому признаку различают файерволы сетевого, сеансового и прикладного уровней. Если же файервол анализирует и фильтрует трафик на нескольких уровнях, то его относят к самому высокому из всех этих уровней.

Уровень протокола, на котором работает файервол, часто используют в качестве интегральной характеристики, поскольку с ней коррелируют другие признаки файервола. Например, файерволы, работающие на сеансовом и прикладном уровнях, чаще относятся к разряду файерволов с запоминанием состояния, а более простые файерволы сетевого уровня — без запоминания. Далее мы приводим типичные сочетания характеристик файерволов, упорядоченные по уровням.

Управляемые коммутаторы, обладающие расширенным набором функций, в том числе с возможностью фильтрации кадров канального уровня на основе задаваемых администратором списков доступа, могут условно быть отнесены к **файерволам канального уровня**.

Файерволы сетевого уровня, называемые также **файерволами с фильтрацией пакетов** (packet filtering firewall), в полном соответствии со своим названием решают задачу фильтрации пакетов по IP-адресам (как источника, так и приемника), а также по значению поля протокола верхнего уровня — в пакет сетевого уровня могут быть вложены сообщения протоколов TCP, UDP, ICMP и др. Более того, несмотря на свое название, такие файерволы работают и на более высоком, транспортном уровне, то есть на уровне портов TCP и UDP, но только на основе статических правил, при которых не отслеживаются состояния соединений, то есть в режиме без запоминания состояния. Поэтому с помощью файервола сетевого уровня можно заблокировать доступ к определенному приложению, запретив прохождение пакетов с определенными номерами портов TCP или UDP, но нельзя защитить сеть от искаженного сеанса TCP или HTTP, потому что это требует отслеживания последовательности шагов в сеансе и, следовательно, запоминания состояния сеанса, а этого файерволы сетевого уровня делать не умеют.

Этому типу файерволов соответствуют маршрутизаторы, поддерживающие пользовательские фильтры, а также программные персональные файерволы операционных систем. Опытный администратор может задать достаточно изощренные правила фильтрации, учитывающие многие требования, касающиеся защиты ресурсов внутренней сети. Тем не менее этот тип сетевых экранов уступает по степени защиты другим типам. *Преимуществами* файерволов сетевого уровня являются простота, невысокая стоимость и минимальное влияние на производительность сети (то есть их дополнительная работа по фильтрации трафика не замедляет маршрутизацию пакетов между двумя сетями).

Файерволы сеансового уровня являются файерволами с *запоминанием состояний* соединений на уровнях ниже прикладного. Файерволы сеансового уровня эффективно противодействуют тем видам атак на протокол TCP, в которых нарушается *трехшаговая процедура установления соединения*. Мы подробно обсудим этот случай в следующей главе.

Файерволы сеансового уровня используются для защиты и от других типов атак, в том числе таких, для распознавания которых требуется анализ не отдельных пакетов, а их последовательности. Например, атаку Ping flood можно распознать по слишком маленькому интервалу между эхо-запросами от одного и того же источника, для чего устанавливается предельно допустимый минимальный интервал между эхо-запросами, а затем фиксируется время прихода очередного запроса. Если оно оказывается меньше предельного, то пакет отбрасывается. Таким образом, запоминание состояния сеансов может обобщаться и на протоколы, работающие *без установления соединения* (ICMP, UDP, DNS), а это означает, что, в отличие от сетевых файерволов, файерволы сеансового типа способны защитить сеть от некоторых видов DoS-атак, даже если в этих атаках не используется протокол TCP.

Файерволы прикладного уровня способны интерпретировать, анализировать и контролировать *содержимое сообщений*, которыми обмениваются приложения. Они также работают на основе фильтрации с запоминанием состояния, но анализируют состояния не только протоколов нижних уровней (вплоть до транспортного), но и прикладного уровня — таких, как протоколы SSH, HTTP, FTP, SQL, SMTP, POP3, IMAP, FTP, SSH, SQL и др.

ПРИМЕЧАНИЕ

Особым типом файерволов этого уровня является прокси-сервер, который перехватывает запросы клиентов к внешним серверам с тем, чтобы потом отправить их от своего имени. Этот тип сетевых экранов обеспечивает самый высокий уровень защиты, хотя и имеет свои недостатки, например, требует больших вычислительных затрат. Кроме того, прокси-сервер может скрывать адрес «доверившегося» ему клиента, что снижает эффективность других средств защиты.

Следует отличать функции по блокировке приложений, реализуемые файерволами сетевого уровня, от защиты приложений внутренней сети файерволами прикладного уровня. Файервол сетевого уровня понимает структуру заголовков пакетов TCP и UDP, за счет чего может запретить или разрешить прохождение пакетов с определенным номером программного порта TCP или UDP, а так как этот номер присвоен серверной части некоторого приложения, то блокируется *весь трафик извне* к этому приложению.

Файервол прикладного уровня действует более гибко. Он контролирует сеанс некоторого приложения, разрешая или запрещая *определенные виды взаимодействия* между внутренней и внешней частями этого приложения в соответствии с заданными правилами. Так, при контроле веб-службы файервол может разрешить использование только определенных команд протокола HTTP, запретив остальные. В список запрещенных команд могут попасть опасные для веб-сервера команды PUT и DELETE. Аналогично при контроле почтовой службы файервол прикладного уровня может не пропускать вовне письма, не подписанные цифровой подписью отправителя, если это предусмотрено политикой безопасности предприятия.

Файервол прикладного уровня, используемый в качестве корпоративного межсетевого экрана, чаще всего является интегрированным продуктом с модульной структурой, которая позволяет менять набор поддерживаемых функций фильтрации в зависимости от потребностей конкретной сети. За счет дополнительных модулей файерволы прикладного уровня могут поддерживать самые разные функции защиты программного обеспечения, например:

- *трансляция внутренних IP-адресов* пользователей на основе стандарта NAT;
- *антивирусный контроль* загружаемых пользователем файлов и получаемых писем «на лету»;
- *контроль контента*, заключающийся, например, в ограничении доступа пользователей к внешним веб-сайтам, страницы которых содержат заданные ключевые слова; такой же контроль может применяться к электронным письмам, отправляемым вовне;
- *транзитная аутентификация пользователей*, обращающихся к некоторому приложению на внутреннем сервере, — эта функция полезна для тех приложений, которые либо не выполняют аутентификацию пользователей совсем, либо делают это незащищенным способом, как, например, FTP-сервер, который принимает пароли пользователей в открытом виде: файервол перехватывает обращение пользователя к FTP-серверу (команду USER) и организует сеанс логического входа пользователя, например, с сервером аутентификации Kerberos, если именно такой способ аутентификации применяется в корпоративной сети;
- *централизованное шифрование* электронных писем пользователей, что избавляет пользователей от необходимости конфигурировать такую функцию на своих клиентских компьютерах (для этого файервол должен хранить цифровые сертификаты пользователей);
- *функции шлюза VPN* с удаленными подразделениями предприятия и удаленными пользователями.

Программные файерволы хоста

Программные файерволы хоста являются частью его программного обеспечения, реализуя наряду с файерволом сети двухступенчатый контроль трафика. Программный файервол работает в режиме ядра ОС, контролируя сетевые интерфейсы хоста и перехватывая пакеты до передачи их протоколам стека TCP/IP.

Программные файерволы хоста являются, как правило, *файерволами сетевого уровня без запоминания состояния сеанса*. Отслеживание состояния сеанса требует значительных вычислительных ресурсов компьютера, и поддержка файерволом хоста такой функции могла бы привести к существенному замедлению выполнения основных его функций. Как и файерволы сетевого уровня на основе маршрутизаторов, программные файерволы хоста позволяют применять правила, учитывающие номера портов TCP/UDP. Это означает, что пользователь хоста может разрешать или запрещать доступ по сети к определенным приложениям хоста, пользующимися закрепленными за ними портами. Посмотрим, как можно блокировать доступ по сети к приложениям с помощью программного файервола iptables, имеющегося практически во всех версиях Unix/Linux. Этот файервол запускается как Unix-демон и работает на основе правил, записанных в текстовом виде в файле /etc/sysconfig/iptables. Правила состоят из трех секций:

- `INPUT` — правила фильтрации входящего трафика;
- `OUTPUT` — правила фильтрации исходящего трафика;
- `FORWARD` — правила фильтрации транзитного трафика в том случае, когда хост работает как IP-маршрутизатор, имея два сетевых интерфейса.

На рис. 30.3 показан пример правил секции INPUT. Как можно заметить, их синтаксис похож на синтаксис правил маршрутизаторов Cisco (см. начало главы).

Рассмотрим, например, такое правило:

```
ACCEPT tcp - - anywhere anywhere state NEW tcp dpts:vnc-server:5903
```

```
[root@ganymede sysconfig]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
ACCEPT     udp  --  anywhere             anywhere            udp dpt:domain
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:domain
ACCEPT     udp  --  anywhere             anywhere            udp dpt:bootps
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:bootps
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
ACCEPT     icmp --  anywhere             anywhere
ACCEPT     all  --  anywhere             anywhere
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpts:vnc-server:5903
ACCEPT     udp  --  anywhere             anywhere            state NEW udp dpts:vnc-server:5903
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:oa-system
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:webcache
ACCEPT     udp  --  anywhere             anywhere            state NEW udp dpt:webcache
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:pcsync-https
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:msgsrvr
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ddi-tcp-1
REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited
```

Рис. 30.3. Правила секции INPUT файервола iptables

Оно говорит о том, что пакеты, удовлетворяющие условию правила, должны быть приняты (`ACCEPT`). Этому правилу удовлетворяют пакеты протокола TCP (`tcp`) с любым адресом источника (`anywhere`) и любым адресом приемника (`anywhere`), относящиеся к новому сеансу TCP (`state NEW`, то есть к пакетам с признаком SYN) и имеющие в поле порта назначения значения в диапазоне от 5900 (это стандартный порт сервиса vnc-server, поэтому порт задан с помощью своего имени) до 5903.

Список включает еще несколько аналогичных правил, а завершает его правило `REJECT all anywhere anywhere`, запрещающее все, что не разрешено явно.

Влияние DHCP на работу файервола

Назначение IP-адресов может происходить вручную в результате выполнения процедуры конфигурирования интерфейса либо автоматически с помощью протокола динамического конфигурирования хостов DHCP. Этот протокол работает в локальных сетях, так как основан на широковещательных запросах клиентов к DHCP-серверам.

В сети, где адреса назначаются динамически, нельзя быть уверенными в адресе, который в данный момент имеет тот или иной узел. Такое непостоянство IP-адресов влечет за собой некоторые проблемы, прежде всего в работе DNS-серверов и файерволов. Как отмечено, фильтрация трафика во многих случаях происходит на основе IP-адреса защищаемого хоста, а при использовании DHCP этот адрес выдается хосту временно, например на сутки, после чего процедура назначения адреса повторяется, причем нет никакой гарантии, что адрес останется тем же. Поэтому в локальной сети, защищенной файерволом и использующей протокол DHCP, возможны два подхода:

- Применение DHCP только для конфигурирования *клиентских* компьютеров, для которых обычно не требуется отображение доменного имени в IP-адрес, и конфигурирование серверов с постоянными, статическими IP-адресами.
- Применение DHCP и для клиентов, и для серверов, но настройка сервера DHCP производится таким образом, чтобы он для серверов создавал *статические записи* DHCP, в которых MAC-адрес запроса связывается с IP-адресом ответа сервера. При поступлении запроса с MAC-адресом, имеющимся в какой-либо статической записи сервера DHCP, последний помещает в ответ IP-адрес из этой статической записи, а не из динамического пула IP-адресов.

Прокси-серверы

Функции прокси-сервера

Прокси-сервер (proxy server) — особый тип приложения, выполняющего функции посредника между клиентскими и серверными частями распределенных сетевых приложений, причем предполагается, что клиенты принадлежат внутренней (защищаемой) сети, а серверы — внешней (потенциально опасной) сети. Роль транзитного узла позволяет прокси-серверу логически разорвать прямое соединение между клиентом и сервером с целью контроля процесса обмена сообщениями между ними.

Подобно файерволу, прокси-сервер может эффективно выполнять свои функции только при условии, что контролируемый им трафик не пойдет обходным путем.

Прокси-сервер может быть установлен не только на платформе, где работают все остальные модули файервола (рис. 30.4, *а*), но и на любом другом узле внутренней сети или сети демилитаризованной зоны (рис. 30.4, *б*). В последнем случае программное обеспечение клиента должно быть сконфигурировано таким образом, чтобы у него не было возможности установить прямое соединение с ресурсным сервером, минуя прокси-сервер.

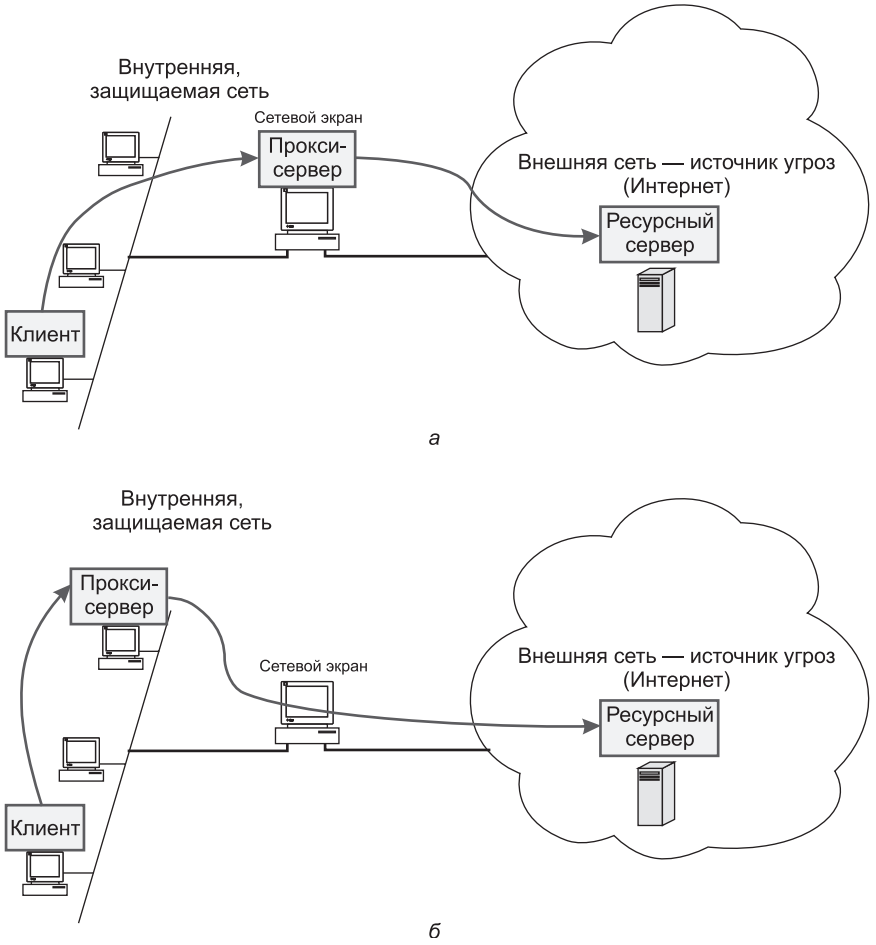


Рис. 30.4. Варианты размещения прокси-сервера

Когда клиенту необходимо получить ресурс от какого-либо сервера (файл, веб-страницу, почтовое сообщение), он посылает свой запрос соответствующему прокси-серверу. Последний анализирует этот запрос и на основании заданных ему администратором правил решает, каким образом он должен быть обработан: отброшен, передан без изменения ресурсному серверу, модифицирован тем или иным способом перед передачей, немедленно обработан силами самого прокси-сервера и др.

В качестве правил, которыми руководствуется прокси-сервер, могут выступать условия пакетной фильтрации. Правила могут быть достаточно сложными — например, в рабочие часы блокируется доступ к тем или иным узлам и/или приложениям, а доступ к другим узлам разрешается только определенным пользователям, причем для FTP-серверов пользователям разрешается делать лишь загрузку, тогда как выгрузка запрещается. Прокси-серверы могут также фильтровать почтовые сообщения по типу пересылаемого файла (например, запретить получение приложений формата MP3) и по их контенту. К разным

пользователям могут применяться разные правила фильтрации, поэтому часто на прокси-серверы возлагается и задача аутентификации пользователей. Если после всесторонней оценки запроса от приложения прокси-сервер констатирует, что запрос удовлетворяет условиям прохождения дальше во внешнюю сеть, то он по поручению приложения-клиента, но от своего имени выполняет процедуру соединения с сервером, затребованным данным приложением.

Помимо основных функций, многие прокси-серверы могут выполнять другие полезные операции, например, *обнаруживать вирусы* еще до того, как они попали во внутреннюю сеть, или *собирать статистические данные* о работе пользователей сети в интернете. В некоторых случаях прокси-сервер может изменять запрос клиента. Например, если в него встроена функция трансляции сетевых адресов (см. далее раздел «Трансляция сетевых адресов»), то он может в пакете запроса *подменять IP-адреса* и/или номера портов TCP и UDP отправителя. Поэтому многие атаки, построенные на знании злоумышленником адресов узлов внутренней сети, становятся нереализуемыми.

Прокси-сервер, выступая посредником между клиентом и сервером, взаимодействующими по определенному протоколу, не может не учитывать специфику этого протокола. Так, для каждого из протоколов HTTP, HTTPS, SMTP/POP, FTP, telnet существует особый прокси-сервер, ориентированный на использование соответствующими приложениями: веб-браузером, программой электронной почты, FTP-клиентом, клиентом telnet. Каждый из этих посредников принимает и обрабатывает пакеты только того типа приложений, для обслуживания которого он был создан. Обычно несколько разных прокси-серверов объединяют в один программный продукт.

Посмотрим, как учитывает специфику протокола *прокси-сервер, ориентированный на веб-службу*. Этот тип прокси-сервера может, например, выполнить собственными силами запрос веб-клиента, не отсылая его к соответствующему веб-серверу. Работая транзитным узлом при передаче сообщений между браузерами и веб-серверами интернета, прокси-сервер не только передает клиентам запрашиваемые веб-страницы, но и сохраняет их в своей *кэш-памяти* на диске. В соответствии с алгоритмом кэширования на диске прокси-сервера оседают наиболее часто используемые веб-страницы. При получении запросов к веб-серверам прокси-сервер прежде всего проверяет, есть ли запрошенная страница в его кеше. Если есть, то она немедленно передается клиенту, а если нет, то прокси-сервер обычным образом делает запрос от имени своего доверителя. Прокси-сервер веб-службы может осуществлять *административный контроль* проходящего через него контента, в частности, ограничивать доступ клиента к сайтам, имеющим IP-адреса или DNS-имена из «черных списков». Более того, он может *фильтровать сообщения* на основе ключевых слов. Различают прокси-серверы прикладного и сеансового уровней:

- **Прокси-сервер прикладного уровня**, как это следует из его названия, умеет «вклиниваться» в процедуру взаимодействия клиента и сервера по одному из прикладных протоколов, например HTTP, HTTPS, SMTP/POP, FTP или telnet. Чтобы выступать в роли посредника на прикладном уровне, прокси-сервер должен понимать смысл команд, знать форматы и последовательность сообщений, которыми обмениваются клиент и сервер соответствующей службы. Это позволяет прокси-серверу проводить анализ содержимого сообщений, делать заключения о подозрительном характере того или иного сеанса.
- **Прокси-сервер сеансового уровня** выполняет свою посредническую миссию на транспортном уровне, контролируя TCP-соединение. Очевидно, что, работая на более

низком уровне, прокси-сервер обладает гораздо меньшим интеллектом и имеет меньше возможностей для выявления и предупреждения атак. Но он обладает и очень важным преимуществом перед прокси-сервером прикладного уровня — *универсальностью*, то есть он может быть использован любыми приложениями, работающими по протоколу TCP (а в некоторых случаях и UDP).

«Проксификация» приложений

Список приложений (точнее, их клиентских частей), которые должны передавать свои запросы во внешнюю сеть исключительно через прокси-сервер, определяется администратором. Чтобы эти приложения поддерживали такой режим выполнения, их программы должны быть соответствующим образом написаны.

Приложения должны быть оснащены средствами, которые распознавали бы запросы к внешним серверам и перед отправкой преобразовывали эти запросы так, чтобы все они попадали на соответствующий прокси-сервер, а не передавались в соответствии со стандартным протоколом прямо на сервер-адресат.

Эти средства должны также поддерживать *протокол обмена сообщениями приложения-клиента с прокси-сервером*. В последние годы в большинстве приложений, ориентированных на работу через интернет, предусмотрена встроенная поддержка прокси-сервера. Такой поддержкой, например, оснащены все веб-браузеры и все клиенты электронной почты, которыми мы сейчас пользуемся.

«Проксификация» приложения, изначально не рассчитанного на работу через прокси-сервер, требует изменения исходного кода с последующей перекомпиляцией — очевидно, что такая работа не представляет сложностей для разработчиков данного приложения, но администратор сети не всегда может ее выполнить, например, из-за отсутствия исходного кода или же необходимой квалификации программиста. Задача администратора заключается в приобретении готовых приложений, совместимых с используемым в сети прокси-сервером. Однако даже приобретение готового «проксифицированного» клиента не делает его готовым к работе — необходимо еще конфигурирование, в частности, нужно сообщить клиенту адрес узла сети, на котором установлен соответствующий прокси-сервер.

Как можно было ожидать, процедура «проксификации» значительно упрощается для прокси-сервера сеансового уровня. Для «проксификации» приложения в этом случае достаточно внести простейшие исправления в исходный текст, которые сводятся к замене всех *стандартных вызовов сетевых функций* версиями этих функций из библиотеки процедур соответствующего прокси-сервера, а затем выполнить перекомпиляцию его программы.

Еще один подход к «проксификации» — встраивание поддержки прокси-сервера в операционную систему. В этом случае приложения могут оставаться в полном неведении о существовании в сети прокси-сервера — все необходимые действия за них выполнит ОС.

Трансляция сетевых адресов

Маршрутизация осуществляется на основе адресов назначения, которые помещены в заголовки пакетов и, как правило, остаются неизменными с момента их формирования отправителем до момента поступления на узел получателя. Но из этого правила есть и ис-

ключения. Например, в широко применяемой сегодня технологии **трансляции сетевых адресов** (Network Address Translation, **NAT**) предполагается продвижение пакета во внешней сети (в интернете) на основании адресов, отличающихся от тех, которые используются для маршрутизации пакета во внутренней (корпоративной) сети.

Одной из причин использования технологии NAT является *дефицит IPv4-адресов*. Если по каким-либо причинам предприятию, у которого имеется потребность подключиться к интернету, не удается получить у поставщика услуг необходимое количество глобальных IPv4-адресов, то оно может прибегнуть к технологии NAT. В этом случае для адресации внутренних узлов служат специально зарезервированные для этих целей **частные адреса**. Мы уже обсуждали их в главе 13. Чтобы узлы с частными адресами могли связываться через интернет между собой или с узлами, имеющими глобальные адреса, необходимо использовать технологию NAT.

Технология NAT также оказывается полезной, когда предприятие из соображений *безопасности* желает скрыть адреса узлов своей сети, чтобы не дать возможности злоумышленникам составить представление о структуре и масштабах корпоративной сети, а также о структуре и интенсивности исходящего и входящего трафиков.

Именно технология NAT стояла у истоков зарождения файерволов как отдельного класса продуктов. В начале 90-х годов, когда дефицит адресов IPv4 еще мало ощущался, несколько специалистов основали компанию Network Translation и разработали программный продукт PIX, который позволял транслировать сетевые адреса. Позднее эту компанию приобрела компания Cisco, а программный продукт стал знаменитым фаерйволом Cisco PIX Firewall, являющимся одним из флагманов средств защиты этого класса.

Традиционная технология NAT

Технология трансляции сетевых адресов имеет несколько разновидностей, наиболее популярная из которых — **традиционная технология трансляции сетевых адресов** — позволяет узлам из частной сети прозрачным для пользователей образом получать доступ к узлам внешних сетей. Подчеркнем, что в данном варианте NAT решается проблема организации только тех сеансов связи, которые *исходят* из частной сети. Направление сеанса в данном случае определяется положением инициатора: если обмен данными инициируется приложением, работающим на узле внутренней сети, то сеанс называется исходящим (несмотря на то, что в его рамках в сеть могут поступать данные извне)[1].

Идея технологии NAT — в следующем. Пусть сеть предприятия образует тупиковый домен, узлам которого присвоены частные адреса (рис. 30.5). На маршрутизаторе, связывающем сеть предприятия с внешней сетью, установлено программное обеспечение NAT. NAT-устройство динамически отображает набор частных адресов {IP*} на набор глобальных адресов {IP}, полученных предприятием от поставщика услуг и присвоенных внешнему интерфейсу маршрутизатора предприятия.

---

[1] Традиционная технология NAT в виде исключения допускает сеансы обратного направления, заранее выполняя статическое взаимно однозначное отображение внутренних и внешних адресов для некоторого ограниченного набора узлов.

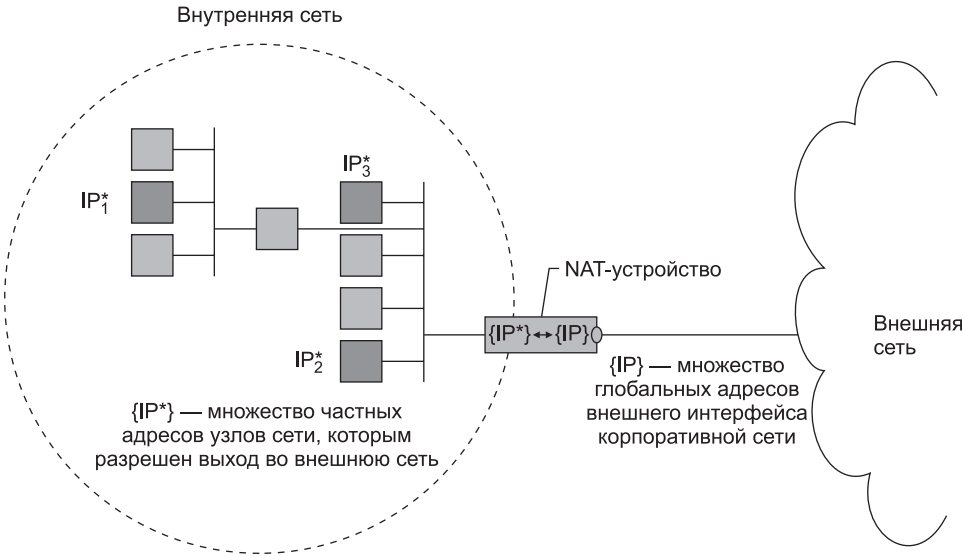


Рис. 30.5. Схема действия традиционной технологии NAT

Важным для работы NAT-устройства является правило распространения маршрутных объявлений через границы частных сетей. Объявления протоколов маршрутизации о внешних сетях пропускаются пограничными маршрутизаторами во внутренние сети и обрабатываются внутренними маршрутизаторами. Обратное утверждение неверно – маршрутизаторы внешних сетей не получают объявлений о внутренних сетях, объявления о них отфильтровываются при передаче на внешние интерфейсы. Поэтому внутренние маршрутизаторы «знают» маршруты ко всем внешним сетям, а внешним маршрутизаторам ничего не известно о существовании частных сетей. Традиционная технология NAT подразделяется на технологии **базовой трансляции сетевых адресов** (Basic Network Address Translation, Basic **NAT**) и **трансляции сетевых адресов и портов** (Network Address Port Translation, **NAPT**). В технологии Basic NAT для отображения используются только IP-адреса, а в технологии NAPT – еще и так называемые транспортные идентификаторы, в качестве которых чаще всего выступают порты TCP и UDP.

Базовая трансляция сетевых адресов

Если количество локальных узлов, которым необходимо обеспечить выход во внешнюю сеть, меньше или равно имеющемуся количеству глобальных адресов, то для каждого частного адреса гарантировано однозначное отображение на глобальный адрес. В каждый момент времени количество внутренних узлов, которые получают возможность взаимодействовать с внешней сетью, ограничивается количеством адресов в глобальном наборе. В этой ситуации целью трансляции является не столько решение проблемы дефицита адресов, сколько обеспечение безопасности.

Частные адреса некоторых узлов могут отображаться на глобальные адреса *статически*. К таким узлам можно обращаться извне, используя закрепленные за ними глобальные адреса. Соответствие внутренних адресов внешним задается таблицей, поддерживаемой

маршрутизатором или другим устройством (например, файерволом), на котором установлено программное обеспечение NAT.

В нескольких тупиковых доменах могут быть совпадающие частные адреса. Например, в сетях *A* и *B* на рис. 30.6 для внутренней адресации применяется один и тот же блок адресов 10.0.1.0/24. В то же время адреса внешних интерфейсов обеих сетей (181.230.25.1/24, 181.230.25.2/24 и 181.230.25.3/24 в сети *A* и 185.127.125.2/24 185.127.125.3/24 185.127.125.4/24

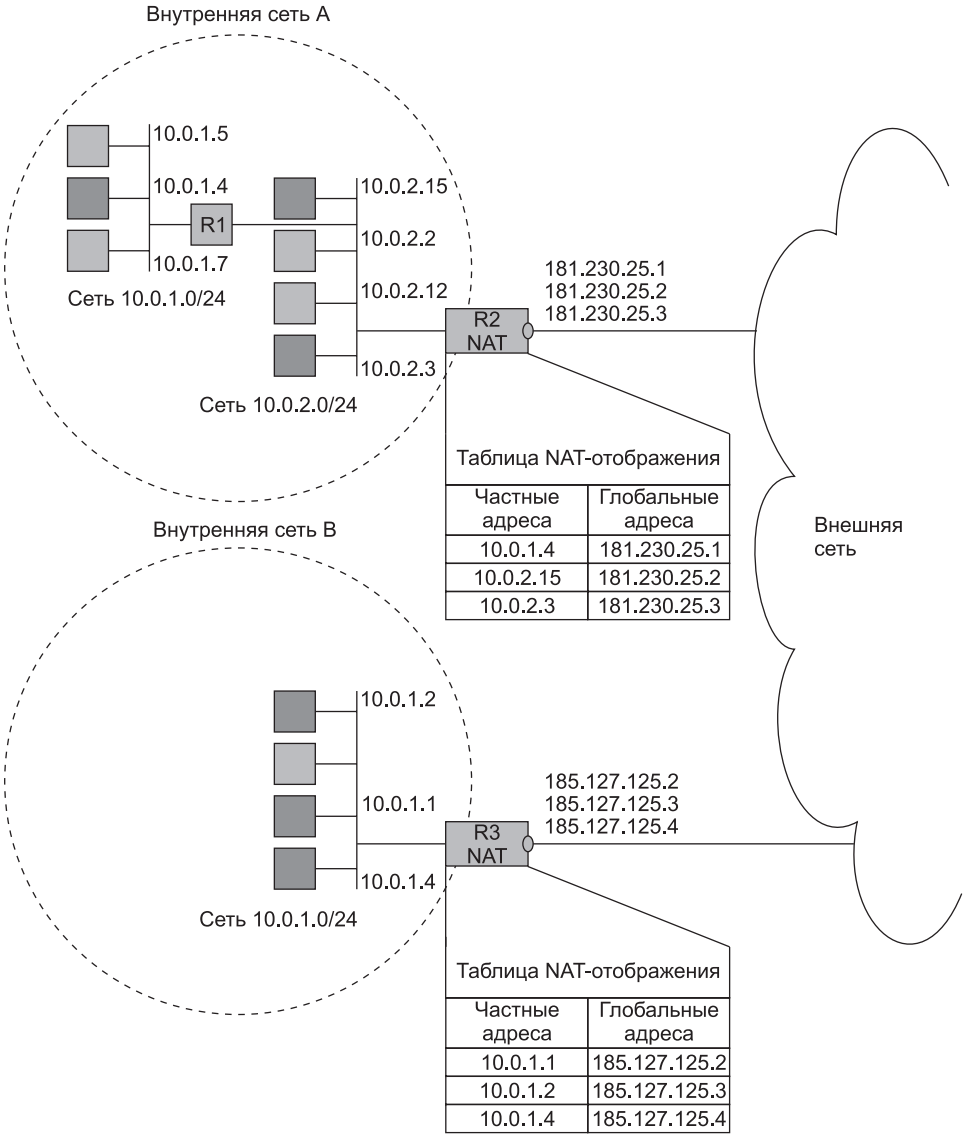


Рис. 30.6. Базовая трансляция сетевых адресов для исходящих сеансов

в сети *B*) уникальны глобально, то есть никакие другие узлы в составной сети их не используют. В данном примере в каждой из сетей только три узла имеют возможность выхода за пределы сети своего предприятия. Статическое соответствие частных адресов этих узлов глобальным адресам задано в таблицах пограничных устройств обеих сетей.

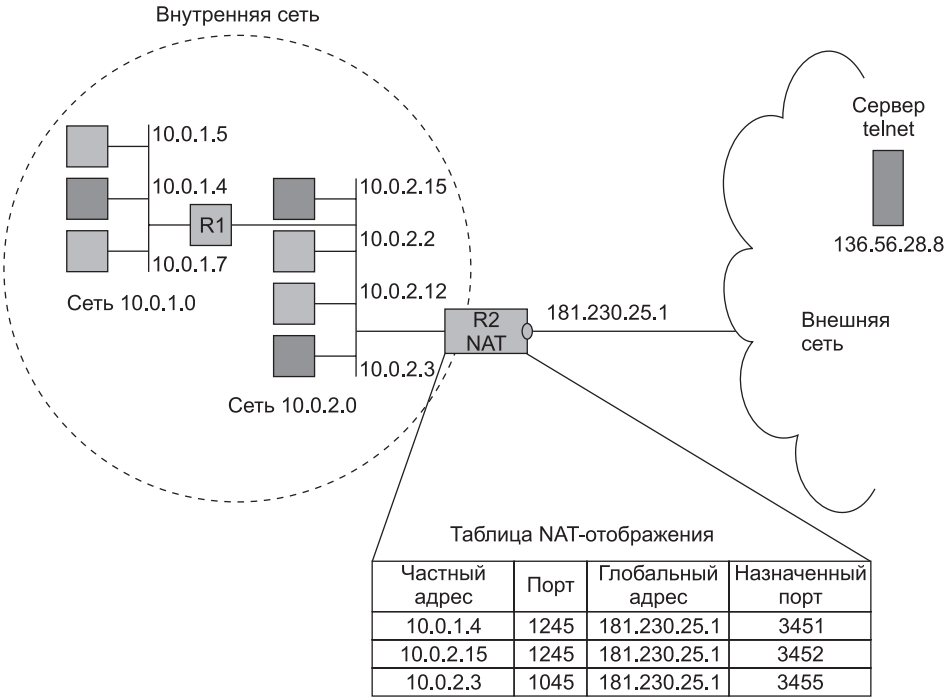
Когда узел 10.0.1.4 сети *A* посылает пакет хосту 10.0.1.2 сети *B*, он помещает в заголовок пакета в качестве адреса назначения глобальный адрес 185.127.125.3/24. Узел-источник по умолчанию направляет пакет своему маршрутизатору R1, которому известен маршрут к сети 185.127.125.0/24. Маршрутизатор передает пакет на пограничный маршрутизатор R2, которому также известен маршрут к сети 185.127.125.0/24. Перед отправкой пакета модуль NAT, работающий на данном пограничном маршрутизаторе, используя таблицу отображения, заменяет в поле адреса источника частный адрес 10.0.1.4 соответствующим ему глобальным адресом 181.230.25.1/24. Когда пакет после путешествия по внешней сети поступает на внешний интерфейс NAT-устройства сети *B*, глобальный адрес назначения 185.127.125.3/24 преобразуется в частный адрес 10.0.1.2. Пакеты, передаваемые в обратном направлении, проходят аналогичную процедуру трансляции адресов.

Заметим, что в описанной операции не требуется участия узлов отправителя и получателя, то есть она прозрачна для пользователей.

Трансляция сетевых адресов и портов

Пусть некоторая организация имеет частную IP-сеть и глобальную связь с поставщиком услуг интернета. Внешнему интерфейсу пограничного маршрутизатора R2 назначен глобальный адрес, остальным узлам сети организации — частные адреса. NAPT позволяет *всем* узлам внутренней сети одновременно взаимодействовать с внешними сетями, используя единственный зарегистрированный IP-адрес. Возникает законный вопрос: как внешние пакеты, поступающие *в ответ* на запросы из частной сети, находят узел-отправитель, если в поле адреса источника всех пакетов, отправляемых во внешнюю сеть, помещается один и тот же адрес — адрес внешнего интерфейса пограничного маршрутизатора?

Для однозначной идентификации узла-отправителя привлекается дополнительная информация. Если в IP-пакете находятся данные протокола UDP или TCP, то в качестве такой информации выступает номер порта UDP или TCP соответственно. Но и это не вносит полной ясности, поскольку из внутренней сети может исходить несколько запросов с совпадающими номерами портов отправителя, а значит, опять возникает вопрос об однозначности отображения единственного глобального адреса на набор внутренних адресов. Решение состоит в том, что при прохождении пакета из внутренней во внешнюю сеть каждой паре {внутренний частный адрес; номер порта TCP или UDP отправителя} ставится в соответствие пара {глобальный IP-адрес внешнего интерфейса; назначенный номер порта TCP или UDP}. Назначенный номер порта выбирается произвольно, однако должно быть выполнено условие его уникальности в пределах всех узлов, получающих выход во внешнюю сеть. Соответствие фиксируется в таблице. Эта модель при наличии единственного зарегистрированного IP-адреса, полученного от поставщика услуг, удовлетворяет требованиям по доступу к внешним сетям для большинства сетей средних размеров. На рис. 30.7 приведен пример, когда в тупиковой сети *A* используются внутренние адреса из блока 10.0.0.0. Внешнему интерфейсу маршрутизатора этой сети поставщиком услуг назначен адрес 181.230.25.1.



Частный адрес	Порт	Глобальный адрес	Назначенный порт
10.0.1.4	1245	181.230.25.1	3451
10.0.2.15	1245	181.230.25.1	3452
10.0.2.3	1045	181.230.25.1	3455

Рис. 30.7. Трансляция сетевых адресов и портов для исходящих сеансов TCP и UDP

Когда хост 10.0.1.4 внутренней сети посылает во внешнюю сеть пакет серверу telnet, то он в качестве адреса назначения использует его глобальный адрес 136.56.28.8. Пакет поступает маршрутизатору R1, который знает, что путь к сети 136.56.0.0/16 идет через пограничный маршрутизатор R2. Модуль NAPT маршрутизатора R2 транслирует адрес 10.0.1.4 и порт TCP 1245 источника в глобально-уникальный адрес 181.230.25.1 и уникально назначенный TCP-порт, в приведенном примере — 3451. В таком виде пакет отправляется во внешнюю сеть и достигает сервера telnet. Когда получатель генерирует ответное сообщение, то он в качестве адреса назначения указывает единственный зарегистрированный глобальный адрес внутренней сети, являющийся адресом внешнего интерфейса NAPT-устройства. В поле номера порта получателя сервер помещает назначенный номер TCP-порта, взятый из поля порта отправителя пришедшего пакета. При поступлении ответного пакета на NAPT-устройство внутренней сети именно по номеру порта в таблице трансляции выбирается нужная строка. По ней определяется внутренний IP-адрес соответствующего узла и действительный номер порта. Эта процедура трансляции полностью прозрачна для конечных узлов.

ПРИМЕЧАНИЕ

Заметьте, что в таблице имеется еще одна запись с номером порта 1245, причем такая ситуация вполне возможна: операционные системы на разных компьютерах независимо присваивают номера портов клиентским программам. Именно для разрешения такой неоднозначности и привлекаются уникально назначенные номера портов.

В технологии NAPT разрешаются только исходящие из частной сети сеансы TCP и UDP. Но бывают ситуации, когда нужно обеспечить доступ к некоторому узлу внутренней сети извне. В простейшем случае, когда служба зарегистрирована, то есть ей присвоен хорошо известный номер порта (например, WWW или DNS), и, кроме того, эта служба представлена во внутренней сети в единственном экземпляре, задача решается просто — служба и узел, на котором она работает, однозначно определяются хорошо известным зарегистрированным номером порта службы.

Завершая рассмотрение технологии NAT, заметим, что помимо традиционной технологии NAT существуют и другие ее варианты, например технология двойной трансляции сетевых адресов, когда модифицируются оба адреса — и источника, и приемника (в отличие от традиционной технологии NAT, когда модифицируется только один адрес). Двойная трансляция сетевых адресов необходима, когда частные и внешние адресные пространства имеют коллизии. Наиболее часто это происходит, когда внутренний домен имеет некорректно назначенные публичные адреса, которые принадлежат другой организации. Подобная ситуация может возникнуть, если сеть организации была изначально изолированной и адреса назначались произвольно, причем из глобального пространства. Или же такая коллизия может быть следствием смены поставщика услуг, причем организация хотела бы сохранить старые адреса для узлов внутренней сети.

Системы мониторинга трафика

Файервол может успешно защитить внутреннюю сеть от разнообразных атак при условии, что его фильтры правильно сконфигурированы. Однако даже правильные фильтры конфигурируются статически, так что для подлинно эффективной защиты требуется *заранее предвидеть* все возможные атаки, что в принципе невозможно. Любой новый тип атаки имеет все шансы «просочиться» через файервол и достичь внутренних серверов защищаемой сети. Обнаружить следы атак, которые смогли преодолеть барьер файервола, можно путем мониторинга сетевого трафика.

Мониторинг сетевого трафика — непрерывный процесс инструментального автоматизированного наблюдения за отдельными параметрами трафика с целью проверки соблюдения SLA, планирования сети, а также предотвращения негативных событий — таких, как технические аварии, угрозы и атаки злоумышленников.

Здесь мы рассмотрим следующие средства мониторинга сетевого трафика:

- *анализаторы протоколов*, или *сетевые сниферы*, позволяют захватывать трафик локальных сетей, представлять его в удобном для анализа виде, но собственно анализ данных оставляют администратору;
- *маршрутизаторы, поддерживающие протокол NetFlow*, собирают обобщенные данные о трафике глобальных сетей, передавая его для анализа программным системам NetFlow, которые автоматизируют поиск атак и угроз;
- *системы обнаружения вторжений* (Intrusion Detection Systems, IDS) специализируются на автоматическом распознавании вторжений и угроз в прослушиваемом трафике локальных сетей.

Анализаторы протоколов

Анализаторы протоколов способны на основе некоторых заданных оператором логических условий захватывать отдельные пакеты и декодировать их, то есть показывать в удобной для специалиста форме вложенность пакетов протоколов разных уровней друг в друга с расшифровкой содержания полей каждого пакета. Дружественный интерфейс, обычно присущий этому классу устройств, позволяет пользователю выводить результаты анализа интенсивности трафика; получать мгновенную и усредненную статистическую оценку производительности сети; задавать определенные события и критические ситуации для отслеживания их возникновения.

Зеркализация портов и неразборчивый режим

Анализатор протоколов представляет собой либо самостоятельное специализированное устройство, либо персональный компьютер, обычно переносной класса Notebook, оснащенный *специальной сетевой картой* и соответствующим программным обеспечением. Программное обеспечение анализатора протоколов состоит из ядра, поддерживающего работу сетевого адаптера и декодирующего получаемые данные, и дополнительного программного кода, зависящего от типа исследуемой сети. В состав некоторых анализаторов может входить также *экспертная система*, способная выдавать пользователю рекомендации о том, какие эксперименты следует проводить в данной ситуации, что могут означать те или иные результаты измерений, как устранить некоторые виды неисправностей в сети и пр.

Анализатор протоколов подключается к сети точно так же, как обычный узел. В примере на рис. 30.8 анализатор подключен к порту P4 коммутатора. Предположим, предстоит анализировать трафик клиентов сети, проходящий через порт коммутатора P2. Так как сеть сегментирована, трафик порта P2 не появляется на порту P4 и анализатор остается без работы, поскольку нужного трафика на его порту просто нет.

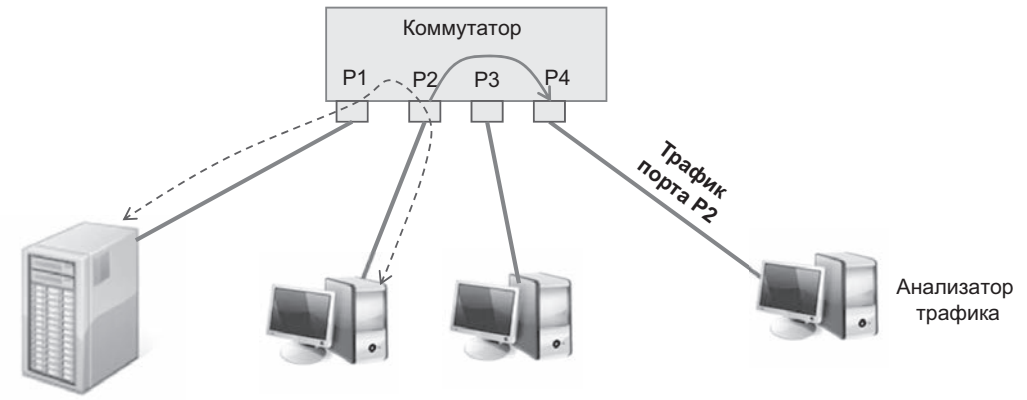


Рис. 30.8. Зеркализация трафика

Для того чтобы анализатор мог работать по назначению, нужно каким-то образом обеспечить прохождение трафика порта P2 через порт P4, например, посредством специальной дополнительной функции, которую поддерживают современные коммутаторы и маршрутизаторы, —

функции **зеркализации портов**. Она состоит в том, что трафик какого-либо порта копируется в буфер другого порта. В нашем примере как входной, так и выходной трафик порта P2 копируется в порт P4 — в таком случае говорят, что порт P2 зеркализирован в порт P4.

Однако мы все еще не достигли поставленной цели. Несмотря на наличие необходимого трафика на порту P4, сетевой адаптер анализатора не принимает появляющиеся на нем кадры, так как они адресованы не ему (у кадров другие MAC-адреса назначения). Для преодоления этого препятствия сетевой адаптер анализатора протоколов должен быть переведен в **режим неразборчивого захвата пакетов** (promiscuous mode). В неразборчивом режиме адаптер захватывает все пакеты, биты которых появляются на его входе.

Анализатор протоколов Wireshark

В качестве примера рассмотрим популярный, свободно распространяемый программный анализатор протоколов **Wireshark**, позволяющий анализировать захваченный трафик, используя иерархическое представление полей пакетов (рис. 30.9).

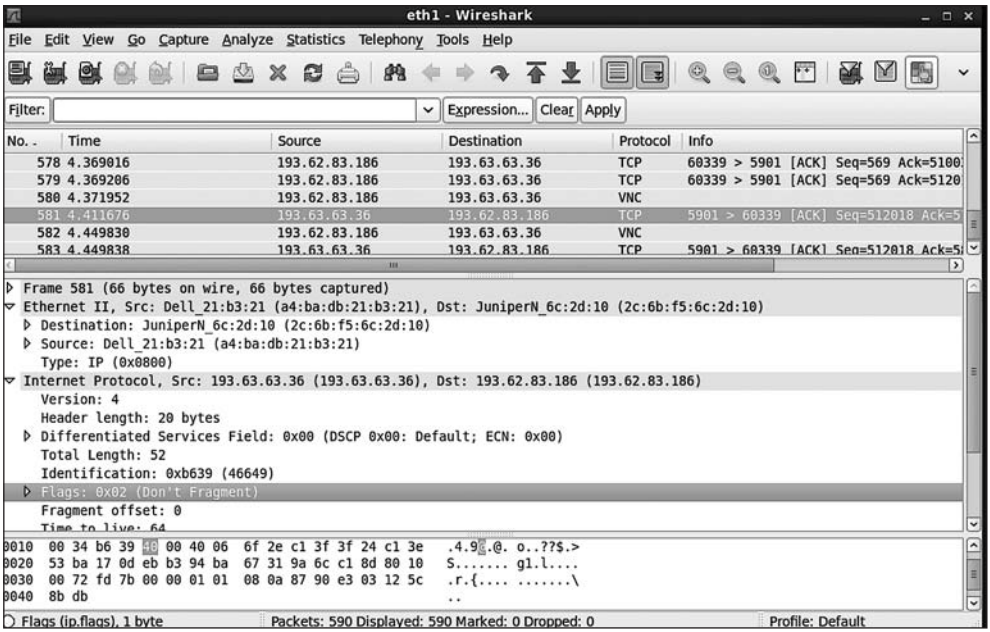


Рис. 30.9. Анализ трафика с помощью программы Wireshark

Верхняя панель окна результатов Wireshark показывает основные параметры каждого захваченного пакета: порядковый номер, время, адреса источника и отправителя и др. Расположенная ниже панель позволяет рассмотреть один из пакетов (в данном случае с номером 581)более детально, при этом поля,состоящие из нескольких подполей, можно раскрывать рекурсивно, добираясь до самого дна иерархии признаков пакета, например до признаков заголовка IP или TCP. Wireshark поддерживает весьма длинный список протоколов от канального до прикладного уровня — на практике это означает возможность раскрытия заголовков протоколовс пояснениями назначения каждого поля.

Wireshark дает возможность гибко задавать фильтры двух типов: *фильтр захвата пакетов* и *фильтр отображения пакетов*; практически любое поле любого протокола может быть использовано в условиях этих фильтров. Захваченные кадры помещаются в файл с расширением .pcap (**p**acket **cap**ture), стандартизованный формат которого сегодня поддерживают практически все программные и программно-аппаратные средства мониторинга трафика.

Применение анализаторов протоколов для обнаружения атак требует значительного опыта, так как за десятками сеансов различных протоколов, часто несущих избыточную информацию, не так-то просто увидеть подозрительную активность. Поэтому часто анализ данных, собранных в файле формата PCAP, автоматизируют с помощью какой-нибудь из доступных *программ анализа трафика*[1]. В то же время предоставляемая анализаторами протоколов принципиальная возможность получить полную картину проходящего через сеть трафика дает шанс специалисту по безопасности разобраться с ситуацией вручную и обнаружить атаку даже в том случае, когда атака является совершенно новой, не известной автоматическим средствам анализа трафика.

(S) *Анализатор протоколов Tcpdump*

Система мониторинга NetFlow

Система мониторинга NetFlow является сегодня основным средством учета и анализа трафика, проходящего через маршрутизаторы и коммутаторы сети. Поддерживающие протокол NetFlow сетевые узлы не только выполняют свою основную работу (передачу пакетов в соответствии с адресом назначения), но и собирают статистику о проходящих через них потоках данных, периодически отправляя собранную информацию в *коллекторы* для хранения и обработки. Практически все ведущие производители сетевого оборудования поддерживают протокол NetFlow, так что для превращения вашего маршрутизатора в источник информации о проходящем трафике достаточно активизировать на нем систему NetFlow. Собранную статистику можно использовать в том числе и для *распознавания сетевых атак*. NetFlow собирает статистику не об отдельных пакетах, а о *потоках* пакетов, определяя поток как последовательность пакетов, объединенных набором общих признаков, в число которых чаще всего входят:[2]

- IP-адрес источника;
- IP-адрес назначения;
- порт TCP/UDP источника;
- порт TCP/UDP приемника;
- тип протокола, переносимого IP-пакетом (полезно в тех случаях, когда это не TCP или UDP; например, это может быть ICMP или OSPF);
- индекс интерфейса, на который получен пакет;
- качество обслуживания — значения байта ToS/DiffServ.

---

[1] Не путать с анализатором протоколов.

[2] Такой набор параметров потока определен в NetFlow версии 5, являющейся на момент написания книги наиболее распространенной. В последующих версиях, например в версии 9, определено более 50 параметров потока, имеется возможность собирать статистику о протоколах MPLS, BGP, IPv6.

NetFlow собирает разнообразную статистику о потоке: время начала и окончания потока, объем данных, переданных с момента начала потока, средняя скорость передачи данных, ну и, естественно, все параметры, определяющие поток, то есть адреса, порты и т. д. Собранная статистика передается в коллекторы (на один или несколько серверов) по окончании потока или же по истечении определенного периода времени.

Маршрутизатор может собирать данные NetFlow в двух режимах: непрерывном, когда обрабатывается каждый пакет, поступающий в маршрутизатор, и выборочный, когда обрабатывается только каждый *n*-й пакет. Выборочный режим менее надежен для распознавания атак, зато он создает гораздо меньше дополнительной нагрузки на маршрутизатор, а для магистрального маршрутизатора, через который проходят десятки, а иногда и сотни тысяч потоков, это существенно.

Типичная система мониторинга на базе NetFlow включает следующие функциональные компоненты (рис. 30.10):

- **Экспортер потока**, называемый также **сенсором**, агрегирует пакеты в потоки и передает статистические данные об этих потоках в один или несколько коллекторов. Экспортером чаще всего является маршрутизатор или коммутатор, хотя могут быть использованы и отдельно стоящие устройства, получающие данные путем зеркалирования порта коммутатора.
- **Коллектор** отвечает за прием, хранение и предварительную обработку данных о потоках, полученных от экспортера потока. Реализуется одним или несколькими серверами.
- **Программа-анализатор** анализирует полученные данные о потоках с целью распознавания возможных атак или возникновения перегрузок сети, установления состава и тенденций изменения трафика в сети.

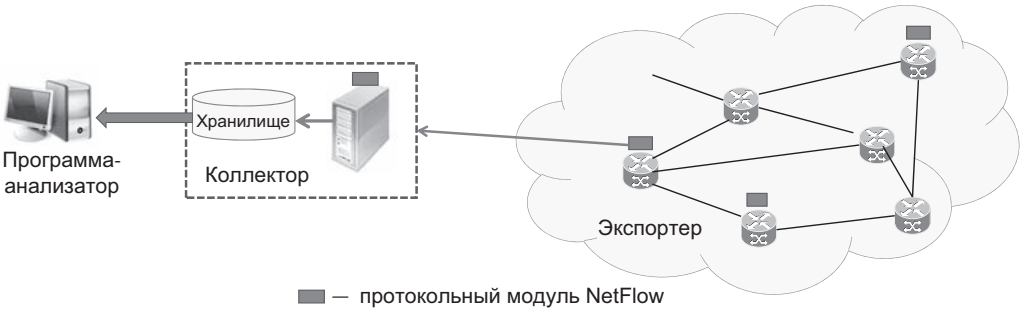


Рис. 30.10. Типичная система мониторинга на базе NetFlow

Важно подчеркнуть, что, в отличие от анализаторов трафика и систем обнаружения атак, NetFlow собирает так называемые **метаданные** о трафике, не заглядывая в поля данных пакетов. Часто статистику NetFlow сравнивают с телефонным счетом, показывающим, с кем и сколько разговаривал данный абонент, но не раскрывающим, о чем он говорил. Однако знания метаданных часто бывает достаточно для того, чтобы распознать атаку. Для этого применяется общий принцип мониторинга сети — сравнение ее текущего поведения с «нормальным», то есть таким, которое устойчиво повторялось в прошлом, и при этом мы знаем, что атак в сети не наблюдалось.

Устойчивые значения статистических характеристик «нормального» поведения сети и ее узлов, которые получены на основании мониторинга сети на довольно значительном периоде времени (недели, месяцы), называются **базовым уровнем** (baseline) характеристик сети.

Другими словами, данные NetFlow служат для поиска аномалий в характере метаданных. Этот же прием используют службы безопасности банков — если, допустим, вы обычно снимаете деньги в банкоматах Москвы, то снятие денег в Рейкьявике является для вас аномалией и ее нужно проверить: может быть, вы просто полетели посмотреть на гейзеры, а может быть, у вас украли данные вашей карточки.

Атака обычно генерирует не совсем обычный трафик, поэтому существуют рекомендации для распознавания таких аномалий. Перечислим основные из них:

- *Выявление узлов с необычно большим числом запросов на установление соединений* (Top N Sessions). Если какой-либо узел вдруг вошел в число *N* узлов, наиболее активных в отношении установления сеансов, то это должно вызывать подозрения (значение *N* обычно выбирается не очень большим, к примеру 10). Такая активность характерна для DoS/DDoS-атак, узлов, зараженных червями, атак сканирования портов и некоторых других видов злоумышленной деятельности. Так, спам-хост будет пытаться отослать как можно больше писем и поэтому устанавливать большое количество соединений в единицу времени с портом 25 (SMTP-порт, на который отправляется почта).
- *Выявление узлов с необычно интенсивным трафиком* (Top N Data). В этом случае хост, который обычно не входил в число *N* самых активных, начинает посылать или получать необычно большое количество данных в единицу времени. Это может быть DoS-атака или же активность червя, пытающегося заразить другие хосты.
- *Анализ SYN и других флагов заголовка TCP.* Наличие необычно большого числа пакетов с установленным флагом SYN или другими флагами заголовка TCP может свидетельствовать о DoS-атаке.
- *Анализ ICMP-сообщений.* Большое количество ICMP-сообщений «Порт/хост/сеть недоступны» может свидетельствовать о сканировании злоумышленником или вирусом хостов и портов.

Другим эффективным методом анализа трафика является проверка значений некоторых полей пакетов на предмет совпадения со значениями, используемыми в известных типах атак (*сравнение с образцами*). Чаще всего образцами атаки являются значения *портов TCP/UDP* и *IP-адресов*. Например, червь SQL Slammer чаще всего использует TCP-порт 1434, а червь W32/Netsky.c всегда использует DNS-сервер с адресом из списка конкретных IP-адресов. Подчеркнем, подход к анализу данных NetFlow должен быть адаптивным, основанным на постоянном обновлении и пополнении базы признаков атак, то есть аналитик должен стараться «идти в ногу» с разработчиками вирусов, ботов и другого вредоносного программного обеспечения.

Системы обнаружения вторжений

Система обнаружения вторжений (Intrusion Detection System, **IDS**) — это программное или аппаратное средство, которое выполняет непрерывное наблюдение за сетевым трафиком и деятельностью субъектов системы с целью предупреждения, выявления и протоколирования атак. В отличие от файерволов, которые строят защиту сети исключительно на

основе анализа сетевого трафика, системы обнаружения вторжений учитывают в своей работе различные подозрительные события, происходящие в системе.

Существуют ситуации, когда сетевой экран оказывается проницаемым для злоумышленника, например, когда атака идет через туннель VPN из взломанной сети, когда инициатором атаки является пользователь внутренней сети и т. п. И дело здесь не в плохой конфигурации межсетевого экрана, а в самом принципе его работы. Несмотря на то что файервол обладает памятью и анализирует последовательность событий, он конфигурируется иа блокирование трафика с *заранее предсказуемыми признаками*, например, по IP-адрссам или протоколам. Так что факт взлома внешней сети с которой у него был установлен защищенный канал и которая прежде вела себя вполне корректно, в правилах экрана отразить нельзя. Точно так же, как и неожиданную попытку легального внутреннего пользователя скопировать файл с паролями или повысить уровень своих привилегий. Подобные подозрительные действия может обнаружить только система, оснащенная агентами, встроенными во многие точки сети, причем она должна следить не только за трафиком, но и за всеми обращениями к критически важным ресурсам отдельных компьютеров, а также обладать информацией о перечне подозрительных действий (сигнатур атак) пользователей. Таковой и является система обнаружения вторжений. Она *не дублирует* действия файервола, а *дополняет* их, производя, кроме того, автоматический анализ всех журналов событий, имеющихся у сетевых устройств и средств защиты, чтобы попытаться найти следы атаки, если ее не удалось зафиксировать в реальном времени.

Другим важным отличием IDS от файерволов является то, что в обязанности IDS не входит блокировка подозрительного трафика. IDS только пытается выявить подозрительную активность и поднять тревогу — обычно путем предупреждения администратора сети электронным сообщением. Кроме поднятия тревоги IDS протоколирует подозрительные пакеты, помещая их в журнал.

Типовая система IDS включает следующие функциональные элементы (рис. 30.11):

- источники данных;
- датчики;
- анализатор;
- администратор;
- оператор;
- менеджер.

Источниками данных для системы обнаружения вторжений являются маршрутизаторы, коммутаторы и хосты локальной сети, словом, все элементы сети, которые передают, генерируют и принимают трафик.

Датчик копирует пакеты, циркулирующие в сети, и передает их анализатору для выявления подозрительной активности. Датчик может представлять собой отдельный компьютер, подключенный к *зеркализованному* порту коммутатора, или же это может быть программный компонент маршрутизатора, который имеет доступ к пакетам, буферизуемым на его интерфейсах. Датчик может осуществлять первичную фильтрацию пакетов, отбирая только те пакеты, которые удовлетворяют некоторым очевидным критериям, например пакеты, направленные к атакуемым наиболее часто публичным веб-серверам.

Анализатор является «мозгом» IDS — он получает данные от датчиков и проверяет их на наличие угроз и подозрительной активности в сети. Анализатор работает на основе правил,

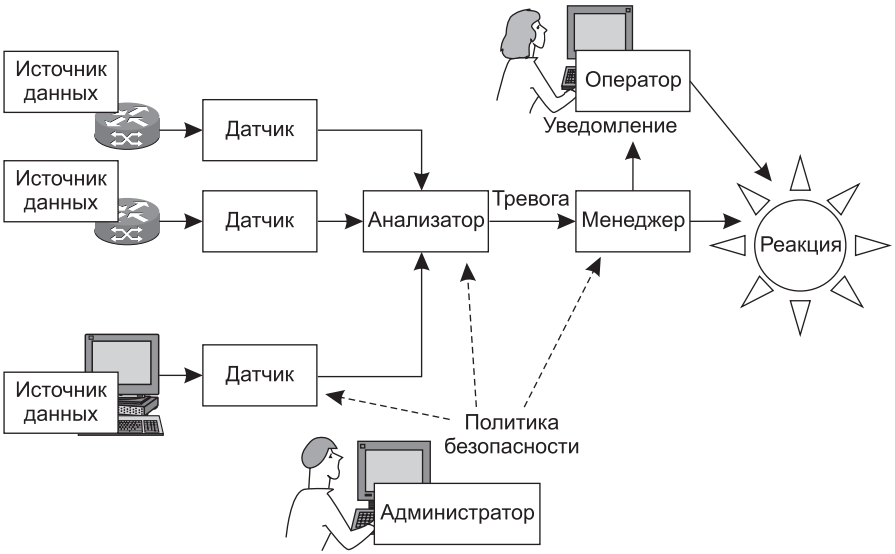


Рис. 30.11. Элементы функциональной архитектуры IDS

составленных *администратором* системы безопасности предприятия в соответствии с политикой безопасности. При выполнении условия одного из правил анализатор вырабатывает сообщение тревоги и передает его *менеджеру* системы IDS — программному компоненту, который хранит конфигурацию IDS и поддерживает удобный интерфейс с оператором IDS. Менеджер IDS оповещает оператора IDS о тревоге в виде некоторого уведомления, привлекающего внимание, например в виде текстовой строки на экране с мерцающим символом, в виде звукового сигнала, продублированного электронным письмом, и т. п.

Оператор системы IDS на основе данных уведомления принимает решение о реакции сети на подозрительную активность — это может быть отключение сетевого интерфейса, через который поступает подозрительный трафик, изменение правил файервола для блокировки определенных пакетов или же игнорирование уведомления, если оператор считает, что вероятность вторжения очень мала. В любом случае все данные о потенциальном вторжении протоколируются в журнале менеджера и могут быть использованы впоследствии для повторного анализа ситуации. Если же IDS выполняет также функции *системы предупреждения вторжений*, то менеджер может автоматически передать команды на маршрутизатор или файервол для блокировки подозрительного трафика.

Описанная схема является функциональной, в реальной системе IDS эти функции не обязательно реализуются в отдельных блоках или модулях системы. В минимальном варианте все функции IDS могут быть сосредоточены в программном обеспечении единственного компьютера, сетевой адаптер которого исполняет роль датчика за счет того, что присоединен к зеркализованному порту коммутатора или маршрутизатора. Правда, в этом случае контролироваться будет только один сегмент корпоративной сети (или несколько сегментов, если на порт коммутатора, к которому подключен компьютер с IDS, зеркализуется несколько рабочих портов коммутатора).

Более масштабируемой является реализация IDS с несколькими датчиками, подключенными к различным сегментам сети и посылающими захваченный трафик центральному анализатору. В качестве таких датчиков может выступать дополнительное программное обеспечение маршрутизатора или коммутатора или же отдельные аппаратные устройства.

Наряду с системами обнаружения вторжений существуют **системы предупреждения вторжений** (Intrusion Prevention Systems, **IPS**), которые выполняют автоматические действия по прекращению атаки в случае ее обнаружения. Часто такие системы перепоручают эту работу файерволу, передавая ему новое правило для блокировки подозрительного трафика.

В IDS для обнаружения вторжений применяются нескольких типов правил:

- *Правила, основанные на сигнатуре (подписи) атаки* (signature rules), используют характерную для той или иной атаки последовательность символов в данных пакета. Например, правило может диктовать поиск строки «user root» в полях FTP-пакета — как известно, этот протокол передает пароли пользователей в открытом виде и применение его суперпользователем root считается грубым нарушением политики безопасности предприятия, так что система IDS должна отслеживать такие случаи. Для эффективной работы система IDS должна иметь обширную постоянно пополняемую базу сигнатур атак.
- *Правила, основанные на анализе протоколов* (protocol rules), связаны с проверкой логики работы протокола и фиксацией отклонений от него. Так как каждый протокол обладает специфической логикой, IDS обычно имеет библиотеку программных модулей, каждый из которых может анализировать поведение определенного протокола. Правила анализа протоколов гораздо сложнее, чем анализа сигнатуры, они требуют высокого быстродействия IDS, чтобы она могла работать в реальном времени.
- *Правила, основанные на статистических аномалиях трафика*, аналогичны тем, которые были рассмотрены в описании технологии анализа данных NetFlow.

Аудит событий безопасности

Аудит (auditing) — это набор процедур учета и анализа всех событий, представляющих потенциальную угрозу для безопасности системы. Аудит является одной из задач, решаемых в целях обеспечения важнейшего требования безопасности — **подотчетности**.

В государственном стандарте подотчетность определяется как свойство безопасной системы, обеспечивающее *«однозначное прослеживание действий любого логического объекта»*. Возможность фиксировать деятельность объектов системы, а затем ассоциировать их с индивидуальными идентификаторами пользователей позволяет выявлять нарушения безопасности и определять ответственных за эти нарушения.

Для обеспечения свойства подотчетности в компьютерных сетях используются различные программно-аппаратные средства, в том числе файерволы, системы обнаружения и предотвращения вторжений, анализаторы протоколов, антивирусные системы, а также некоторые подсистемы ОС.

Аудит позволяет «шпионить» за выбранными объектами и выдавать сообщения тревоги, когда, например, какой-либо рядовой пользователь пытается прочитать или модифицировать системный файл. Если кто-то пытается выполнить действия, отслеживаемые системой

безопасности, то система аудита пишет сообщение в журнал регистрации, идентифицируя пользователя.

Журнал регистрации — это совокупность хронологически упорядоченных записей о специально отобранных событиях. Данные журнала регистрации должны быть надежно защищены от модификации и разрушения неавторизованными субъектами. Таким образом, аудит по своей природе является *реактивным* (а не проактивным) действием, то есть записи становятся достоянием специалиста по безопасности уже по прошествии некоторого времени после того, как события произошли.

Поскольку никакая система безопасности не гарантирует стопроцентную защиту, именно система аудита оказывается последним рубежом в борьбе с нарушениями.

Эту мысль лаконично отражает популярное изречение «Prevention is ideal, but detection is a must!», которое говорит о том, что, бесспорно, предупреждение нарушений — это желательная, но, увы, часто недостижимая цель, в то время как обнаружение и фиксация нарушений — это то, что должно быть сделано обязательно. Действительно, после того как злоумышленнику удалось провести успешную атаку, пострадавшей стороне не остается ничего другого, как обратиться к службе аудита. Если при настройке службы аудита были правильно заданы события, которые требуется отслеживать, то подробный анализ записей в журнале может дать много полезной информации, которая, возможно, позволит найти злоумышленника или, по крайней мере, предотвратить повторение подобных атак устранением уязвимых мест в системе защиты. Аудит действует и как *угроза* потенциальным нарушителям, предупреждая их о том, что в случае несанкционированных действий они легко могут быть выведены на чистую воду.

Функциональный компонент аудита, обеспечивающий *непрерывное* наблюдение за параметрами системы, называется подсистемой **мониторинга**. Мы уже обсуждали мониторинг сетевого трафика. Кроме него объектами мониторинга могут выступать загрузка процессора, статус сетевого интерфейса (активный или пассивный), включенное или выключенное устройство печати.

Ведение журнала событий, особенно в режиме мониторинга, может потребовать слишком много ресурсов вычислительной системы (дискового пространства, вычислительной мощности процессора), а также определенных затрат рабочего времени персонала. Поэтому важно соблюдать баланс между количеством различных видов событий, подлежащих регистрации, с одной стороны, и затрачиваемыми на их протоколирование ресурсами и возможностью их анализа — с другой.

Задачу формирования правил *отбора событий*, подлежащих регистрации, выполняет администратор сети. В ОС, например, к числу таких событий относят попытки успешного и неуспешного логического входа в систему, запуска программ, доступа к защищаемым ресурсам, изменения атрибутов объектов и полномочий пользователей и др. Практически всегда фиксируются события, важные для ОС в целом: рестарт системы, очистка журнала регистрации событий, изменение системного времени и др.

Даже самые простые журналы событий содержат такое огромное количество сведений, что их практически невозможно анализировать «вручную», без специальных средств. Для автоматизации обработки и анализа данных журнала регистрации могут быть использованы различные программные средства, в том числе *средства предварительной обработки данных аудита,* предназначенные для сжатия информации журнала регистрации за счет удаления из него малоинформативных записей, которые только создают ненужный «шум».

При реализации аудита в больших сложных системах, состоящих из множества подсистем с собственными средствами протоколирования событий, иногда необходимо приложить специальные усилия по «синхронизации» журналов регистрации. В частности, поскольку в разных частях большой системы одни и те же объекты могут иметь разные имена и, напротив, разные объекты — одинаковые имена, администраторам, возможно, придется принять специальное соглашение об однозначном именовании объектов.

Типовые архитектуры сетей, защищаемых файерволами

Логическая сегментация защищаемой сети

Мы рассмотрели функциональные возможности файерволов по защите одной сети от возможных атак, исходящих от другой сети. В простейшем случае первая сеть — это единственная внутренняя сеть предприятия, а внешняя представлена всеми сетями интернета, соединенными с внутренней сетью через единственную линию связи предприятия с провайдером интернета. В реальности ситуация оказывается сложнее — сеть предприятия может состоять из нескольких сетей, при этом серверы и хосты этих сетей нуждаются в защите различного типа. Например, если в одной сети находится почтовый сервер и веб-сервер предприятия, а в другой — сервер базы данных клиентов предприятия, то доступ к ним должен регулироваться в соответствии с разными правилами. Если добавить к этому, что многие предприятия соединяют свои сети с интернетом несколькими линиями связи и, возможно, через нескольких провайдеров, то защита сети предприятия приобретает еще одно измерение — защиту всего периметра сети с помощью нескольких файерволов, при этом их правила защиты должны быть согласованными. Под **сетью периметра** понимается совокупность всех связей корпоративной сети с внешними сетями — сетями провайдеров или корпоративными сетями других предприятий.

Для надежной и эффективной защиты корпоративной сети она должна быть *логически сегментирована* таким образом, чтобы ресурсы каждой подсети в отношении мер защиты были подобными. Ресурсы корпоративной сети, к которым обращаются внешние пользователи, безусловно, составляют в отношении мер безопасности отдельную группу — в нее входят почтовый сервер, веб-сервер, DNS-сервер. Повсеместной практикой является выделение таких ресурсов в отдельную группу и размещение их в подсети, которая получила название **демилитаризованной зоны**[1] (demilitarized zone, **DMZ**). В каком-то смысле зона DMZ подобна транзитной зоне аэропорта, потому что пассажирам разрешается использовать только ресурсы этой зоны, а доступ к внутренним ресурсам авиапредприятия им закрыт.

Рассмотрим особенности организации защиты DMZ на примере сети, показанной на рис. 30.12. В этой сети на рубеже защиты установлено два маршрутизатора, между которыми располагается демилитаризованная зона. Маршрутизаторы играют роль файерволов сетевого уровня. В данном случае сеть DMZ является и сетью периметра, так как только она соединяет внутреннюю сеть предприятия с внешними сетями.

[1] Иногда термины «сеть периметра» и «демилитаризованная зона» используются как синонимы.

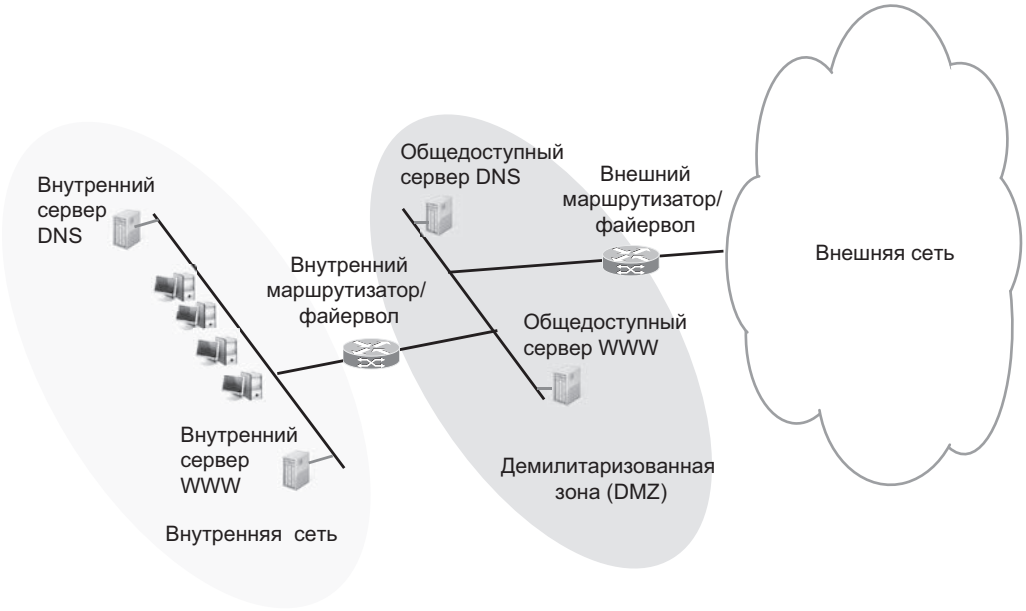


Рис. 30.12. Файервол на базе двух маршрутизаторов

В сети DMZ расположены два общедоступных сервера — внешний DNS-сервер и внешний веб-сервер предприятия. В этой зоне могут быть размещены также прокси-серверы. Учитывая, что само назначение этих компьютеров предполагает практически никак не ограничиваемый доступ к ним внешних пользователей (а значит, и злоумышленников), их необходимо защищать особенно тщательно. Главными задачами при защите этих компьютеров (называемых иногда **компьютерами-бастионами**) является обеспечение целостности и доступности размещенных на них данных для пользователей внешней сети. Эту задачу решают «индивидуальные» средства защиты, устанавливаемые на компьютерах-бастионах: антивирусные программы, фильтры спама и т. п. Кроме того, каждый сервер, к которому разрешено обращение внешних пользователей, должен быть сконфигурирован на поддержку минимально необходимой функциональности. Например, публичный DNS-сервер предприятия не должен быть открытым для любых запросов, так как он может стать инструментом DDoS-атаки.

Чтобы пояснить, каким образом сеть периметра усиливает защиту внутренней сети, посмотрим, что произойдет, если какой-либо злоумышленник сможет взломать первый рубеж защиты — внешний маршрутизатор — и начнет прослушивать трафик подключенной к нему сети периметра. Очевидно, что он получит доступ только к трафику общедоступных серверов, который не является секретным.

Внешний маршрутизатор призван фильтровать трафик с целью защиты сети периметра и внутренней сети. Однако строгая фильтрация в этом случае оказывается невостребованной. Общедоступные серверы по своей сути предназначены для свободного доступа. Что касается защиты внутренней сети, правила фильтрации для доступа к ее узлам и сервисам являются одними и теми же для обоих маршрутизаторов, поэтому внешний маршрутизатор может просто положиться в этом деле на внутренний маршрутизатор. Обычно внешний

маршрутизатор находится в зоне ведения провайдера, и администраторы корпоративной сети ограничены в возможностях его оперативного реконфигурирования. Это является еще одной причиной, по которой функциональная нагрузка на внешний маршрутизатор обычно невелика.

Основная работа по обеспечению безопасности локальной сети возлагается на внутренний маршрутизатор, который защищает ее как от внешней сети, так и от сети периметра. Правила, определенные для узлов сети периметра по доступу к ресурсам внутренней сети, часто бывают более строгими, чем правила, регламентирующие доступ к этим ресурсам внешних пользователей. Это делается для того, чтобы в случае взлома какого-либо компьютера-бастиона уменьшить число узлов и сервисов, которые впоследствии могут быть атакованы с этого компьютера. Именно поэтому внутренний маршрутизатор должен отбрасывать все пакеты, следующие во внутреннюю сеть из сети DMZ, исключая пакеты нескольких протоколов (например, HTTP, SMTP, DNS), абсолютно необходимых пользователям внутренней сети для обращения к внешним серверам, установленным в сети DMZ или же за пределами корпоративной сети. Для исключения возможности обращения внешних пользователей к серверам внутренней сети можно разрешить пропуск к ней только тех TCP-пакетов, которые относятся к TCP-сеансам, установленным по инициативе внутренних пользователей. Например, для маршрутизаторов Cisco это можно сделать с помощью такой строки списка доступа:

```
access-list 200 permit tcp any 201.15.0.0 0.0.255.255 established
```

Здесь 201.15.0.0/16 — диапазон адресов внутренней сети, а список доступа применяется во входном направлении к интерфейсу внутреннего маршрутизатора, к которому подключена сеть DMZ.

Защиту внутренней сети можно усилить, если в ней имеются аналоги внешних серверов, то есть в нашем примере это веб-сервер и DNS-сервер. В подобной конфигурации только этим серверам в случае необходимости разрешается взаимодействовать с серверами зоны DMZ, внутренние же пользователи работают напрямую лишь с внутренними серверами. Например, DNS-сервером по умолчанию для пользователей сети должен быть назначен внутренний DNS-сервер, и только ему позволено изнутри обращаться к внешнему DNS-серверу в том случае, когда он не может разрешить запрос самостоятельно.

Защиту внутренних серверов можно усилить за счет использования частных IP-адресов во внутренней сети. В этом случае внутренний маршрутизатор при трансляции частных адресов должен поддерживать режим NAPT на своем интерфейсе, связывающем его с сетью DMZ.

Архитектура сети с защитой периметра и разделением внутренних зон

Демилитаризованная зона является практически обязательным элементом защищенной архитектуры любой корпоративной сети, однако в общем случае такая сеть должна быть разбита на большее число сегментов со сходными требованиями к защите на основе фильтрации и анализа трафика. Этот подход иллюстрируется архитектурой сети, показанной на рис. 30.13, — достаточно крупная корпоративная сеть разделена на шесть сегментов, каждый из которых представляет отдельную IP-сеть.

Каждый из сегментов сети представляет собой отдельную зону безопасности. Сети зон соединены друг с другом через *корпоративный файервол*, непосредственной связи между этими сетями нет — такая архитектура позволяет надежно реализовывать правила политики безопасности для каждой зоны. Корпоративный файервол выполнен в виде двух устройств, работающих в режиме горячего резервирования, когда каждое из устройств реализует одни и те же правила фильтрации трафика, и в случае отказа одного из устройств работоспособное устройство без разрыва имеющихся соединений может продолжить обслуживать трафик, проходивший ранее через отказавшее устройство.

Корпоративный файервол также контролирует интернет-трафик предприятия, который проходит через две линии связи с различными интернет-провайдерами — такое соединение достаточно типично для крупных корпоративных сетей, так как оно обеспечивает высокую надежность интернет-связи.

Посмотрим на состав и требования к защите каждой из зон.

Зона 1 представляет собой демилитаризованную зону предприятия с открытыми для публичного доступа серверами. Ее особенности мы уже рассмотрели. Иногда эту зону разделяют на две зоны, выделяя веб-серверы в отдельную зону, так как защита веб-сервисов на прикладном уровне существенно отличается от защиты почтового сервера или DNS-сервера.

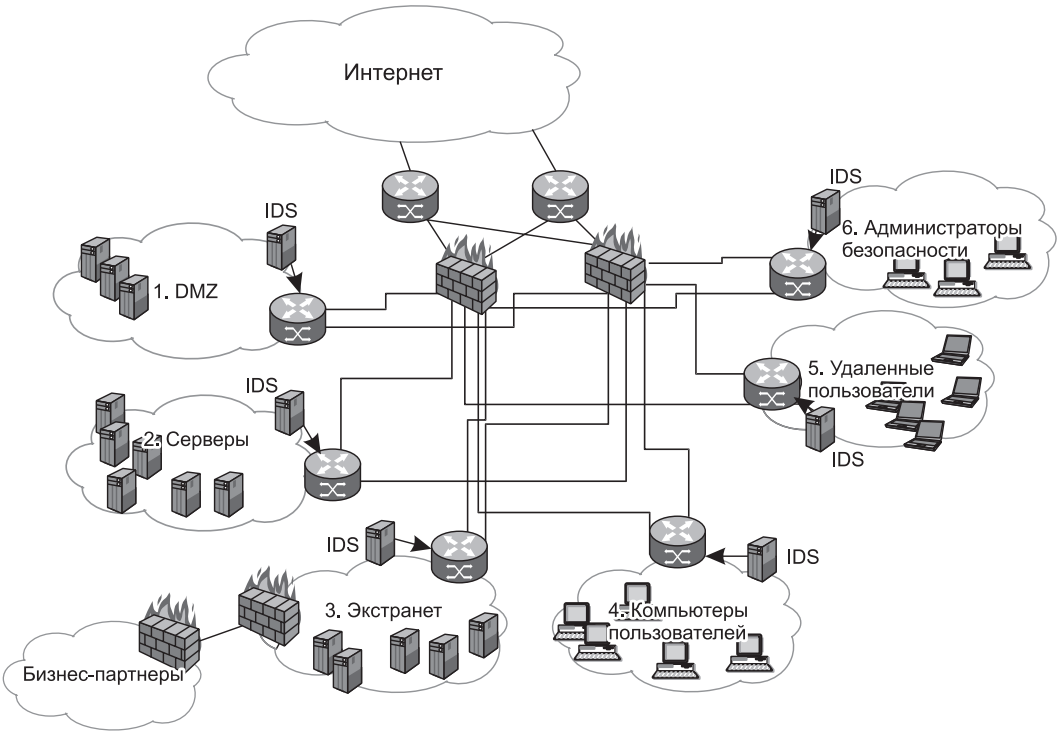


Рис. 28.13. Архитектура сети с несколькими зонами защиты

Зона 2 — это зона внутрикорпоративных серверов, иногда называемая корпоративным порталом. Здесь сосредоточены все информационные ресурсы предприятия, к которым

обращаются сотрудники в своей работе: внутренний веб-сервер, серверы баз данных, внутренний почтовый сервер, серверы приложений управления предприятием и т. п. К этой зоне должны иметь доступ только пользователи предприятия, доступ к ней внешних пользователей должен быть заблокирован. Но и для внутренних пользователей доступ к ресурсам этой зоны должен быть ограничен файерволом в соответствии с принципом минимальных привилегий. На уровне файервола он означает, что пользователям должен быть разрешен доступ только к портам тех приложений, которые им нужны в работе, а все остальные порты должны быть файерволом заблокированы.

Зона 3 — это зона *экстранет* (extranet), где сосредоточены ресурсы, содержащие конфиденциальные данные, к которым разрешен доступ только сотрудникам предприятий-партнеров, а доступ из публичного домена интернета — запрещен. Зона экстранет в нашем примере соединена с предприятиями-партнерами отдельной линией связи, возможно, через отдельного провайдера, и контролируется отдельным файерволом. В других случаях доступ к экстранет может проходить через общие для всех зон линии связи с интернетом и контролироваться тем же файерволом. Для обеспечения конфиденциальности данных может быть применена технология *виртуальных частных сетей* — в этом случае файервол должен выполнять также функции VPN-шлюза.

Зона 4 — это внутренняя зона предприятия, в ней находятся клиентские компьютеры сотрудников предприятия. Для этой зоны разрешается установление соединений с корпоративным порталом (зона 2) и внешними серверами интернета. Установление соединений с компьютерами этой зоны извне, то есть из интернета и из любой другой зоны предприятия, запрещается.

Зона 5 объединяет *мобильных сотрудников* предприятия, то есть сотрудников, пользующихся удаленным доступом из дома или из сетей других предприятий или публичных провайдеров интернета (например, из зон Wi-Fi на вокзалах, в аэропортах, кафе и т. п.). Обычно для хостов этой зоны устанавливаются те же правила доступа, что и для пользователей зоны 4, то есть они имеют доступ к ресурсам зоны 2 и могут устанавливать соединения с ресурсами интернета. Для обеспечения конфиденциальности, как и в случае с экстранет, доступ мобильных пользователей осуществляется через защищенные каналы VPN.

Зона 6 объединяет серверы и клиентские компьютеры, используемые для администрирования средств безопасности предприятия. Здесь сосредоточены серверы политики файерволов, антивирусной защиты, приложений обеспечения безопасности, таких как анализаторы трафика.

К сети каждой зоны подключен сервер IDS, который выполняет анализ трафика этой сети (или, по крайней мере, ее наиболее критичных сегментов) и предупреждает оператора систем безопасности о подозрительной активности.

Файерволы корпоративной сети должны быть сконфигурированы так, чтобы их правила отражали политику безопасности предприятия. Собственно, эта политика и должна определять структуризацию ресурсов сети на зоны, приведенный пример — только один из вариантов этой политики, хотя и достаточно типичный. Возможно и другое разбиение ресурсов на зоны — как более детальное, так и более укрупненное. Например, зона 5, объединяющая в нашем примере всех пользователей предприятия, работающих в его локальной сети (то есть не удаленно), может быть разбита на несколько зон с учетом организационной структуры предприятия — так, в отдельную зону безопасности может быть выделен финансовый отдел.

ГЛАВА 31 Атаки на транспортную инфраструктуру сети

Атаки на транспортные протоколы

TCP-атаки

Протокол TCP используется злоумышленниками и как инструмент для организации атак (обычно — атак отказа в обслуживании), и как цель нападения — нарушение TCP-сеанса атакуемого приложения, например, путем подделки сегмента.

Затопление SYN-пакетами

Этот тип DoS-атаки активно применяется злоумышленниками на протяжении многих лет; впервые он был подробно описан (с приведением кода атаки) в 1996 году, и в том же году началось его практическое «применение», продолжающееся по сей день. Атакуемым является конечный узел — как правило, сервер, работающий с клиентами по протоколу TCP.

Атака затоплением SYN-пакетами (SYN Flood) использует уязвимость процедуры установления логического соединения протокола TCP. Как отмечено в главе 15, эта процедура основана на использовании флагов *SYN* и *ACK*, переносимых в заголовке каждого TCP-сегмента (рис. 31.1, *а*). Для реализации атаки злоумышленник организует передачу на сервер массированного потока пакетов с флагом *SYN*, каждый из которых инициирует создание нового TCP-соединения (рис. 31.1, *б*). Получив пакет с флагом *SYN*, сервер выделяет для нового соединения необходимые ресурсы и в полном соответствии с протоколом отвечает клиенту пакетом с флагами *ACK* и *SYN*. После этого, установив тайм-аут, он ожидает от клиента завершающий пакет с флагом *ACK*, который, увы, так и не приходит.

Аналогичным образом создается множество других «недоустановленных» соединений. Обычно ОС сервера имеет лимит на количество одновременно поддерживаемых «недоустановленных» TCP-соединений (глобально или для каждого программного порта отдельно), так как каждое открытое соединение требует выделения памяти ядра ОС для нового **блока TCB** (Transmit Control Block). Этот блок содержит данные о состоянии соединения: сокет клиента, номер ожидаемого сегмента, указатель на положение сегмента в буфере и др. Блок TCB имеет размер от 280 до 1300 байт в зависимости от типа ОС. При достижении лимита ОС начинает отвергать все последующие запросы на установление TCP-соединений и, следовательно, отказывает в обслуживании всем, в том числе легальным клиентам сервера. По истечении тайм-аута ОС удаляет из памяти блоки TCB «недоустановленных» соединений и начинает устанавливать новые соединения повторно.

Для осуществления атаки затоплением SYN-пакетами атакующий должен заблокировать нормальную реакцию своего компьютера на получение от атакуемого сервера сегмента с флагами *SYN/ACK*. Нормальная реакция состоит в том, что в соответствии с протоколом

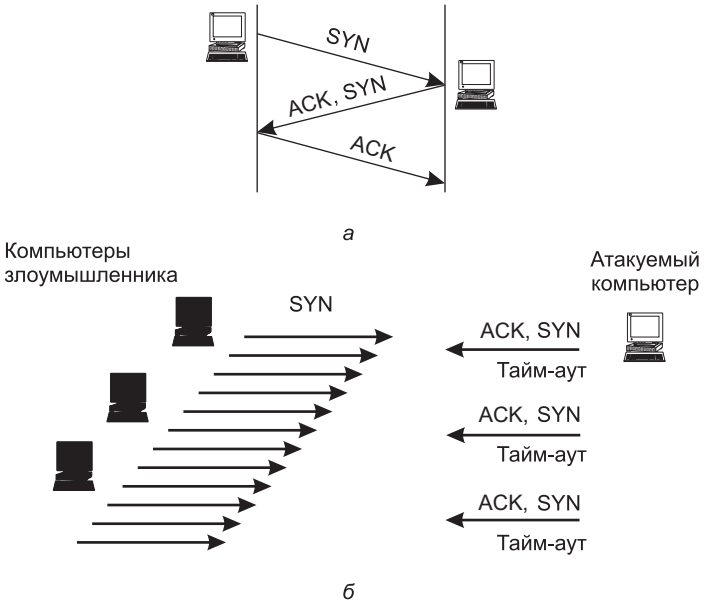


Рис. 31.1. Проведение DoS-атаки, в которой используются особенности протокола TCP: *а* — нормальный порядок установления TCP-соединения; *б* — DoS-атака путем создания множества незакрытых TCP-соединений

TCP атакующий должен отправить в ответ сегмент с флагом *ACK*. Но если это произойдет, то атакуемый сервер посчитает процедуру установления TCP-соединения завершенной, удалит соответствующий блок TCB из списка «недоустановленных» соединений и начнет принимать новые соединения. Таким образом, атака не удастся. Поэтому атакующий фильтрует входящий трафик, отсеивая ответы *SYN/ACK* от атакуемого сервера. Обычно атака затоплением SYN-пакетами обнаруживается посредством выявления в трафике большого количества SYN-сегментов без соответствующего количества ACK-сегментов, идущих от того же источника. При этом заметного всплеска сетевого трафика может и не быть, так как лимит «недоустановленных» соединений сам по себе не столь велик. Главным средством борьбы с атакой затоплением SYN-пакетами является *фильтрация* трафика, поступающего от источника атаки. Для этого нужно определить адрес атакующего узла, что в ряде случаев сделать непросто — атакующий может помещать SYN-сегменты в IP-пакеты с «поддельным» адресом отправителя, применяя *спуфинг*. Спуфинг помогает атакующему преодолеть защитный фильтр и избавиться от вредящих ему ответных SYN/ACK-сегментов атакуемого сервера. Для этого ему достаточно выбрать в качестве «поддельных» адреса, которые не будут реагировать на SYN/ACK-сегменты, — например, адреса несуществующих узлов.

ПРИМЕЧАНИЕ

Спуфинг IP-адресов источника используется во многих типах атак, поэтому борьба с ним — естественный элемент обеспечения сетевой безопасности. Основным средством борьбы является применение на маршрутизаторах техники проверки обратного пути (Reverse Path Check, RPC). Идея этой проверки достаточно проста — пакет должен передаваться маршрутизатором в соответствии с его адресом назначения только в том случае, если его адрес источника имеется в таблице маршрутизации для интерфейса, с которого этот пакет получен. Действительно, если компьютер злоумышленника

подключен к сети 212.100.100.0/24, но генерирует пакеты с адресом источника 25.0.30.18, то маршрутизатор провайдера, к которому подключена сеть 212.100.100.0/24, легко может проверить, что через интерфейс, на который был получен пакет с подделанным адресом, достичь сети 25.0.30.18 нельзя, а значит, пакет нужно отбросить. Однако техника RPC работает не всегда. В тех случаях, когда сеть злоумышленника имеет несколько подключений к сетям разных провайдеров, может произойти отбрасывание легитимных пакетов.

Преодоление атаки путем фильтрации также осложняется, когда поток SYN-сегментов поступает на атакуемый сервер сразу от сотен зараженных компьютеров какой-нибудь сети ботов, то есть когда имеет место **распределенная атака затоплением SYN-пакетами** (DDoS SYN Flood).

Другим способом борьбы с атакой затоплением SYN-пакетами является *изменение параметров протокола TCP* — увеличение предельного числа «недоустановленных» соединений, уменьшение тайм-аута вытеснения старых «недоустановленных» соединений, усложнение логики самой процедуры установления соединения, например, введения специальных *cookie-блоков SYN*. В этом методе при приеме запроса SYN-сервер не запоминает блок TCB в своей оперативной памяти, а посылает его (в сжатом виде) клиенту вместе с SYN/ACK-ответом. При нормальном ходе установления соединения клиент отвечает ACK-сегментом, в котором повторяет сжатый блок TCB. Сервер, получив этот ACK-сегмент, а с ним и все параметры устанавливаемого соединения, создает соответствующий блок TCB в памяти своего ядра. Поскольку в этой модифицированной процедуре на начальном этапе установления соединения ресурсы на сервере не выделяются, то и атака затоплением SYN-пакетами не удается. Разновидностью TCP-атаки затоплением SYN-пакетами является **TCP-атака затоплением ACK-пакетами**, выполняемая путем *отражения*. Злоумышленник посылает SYN-пакеты, в поле адреса источника которых помещен адрес жертвы, на большое количество серверов. Последние отвечают на SYN-пакеты пакетами с установленным битом ACK, «бомбардирующими» атакуемый компьютер и исчерпывающими пропускную способность его входного интерфейса. Этот прием превращает DoS-атаку в DDoS-атаку без использования сети ботов, так как все компьютеры, отвечающие на SYN-запросы, не заражаются предварительно каким-либо вирусом, а работают в полном соответствии со стандартной версией протокола TCP.

Подделка TCP-сегмента

Протокол TCP предназначен для надежной транспортировки сообщений. Для этого каждый сегмент данных сопровождается порядковым номером первого байта сегмента, причем начальные значения этих номеров для каждой из двух сторон, обменивающихся данными, выбирается случайным образом. При приеме очередного сегмента протокол TCP проверяет, находится ли его порядковый номер в пределах окна приема, и только в случае положительного результата такой проверки добавляет принятые данные к байтам, принятым ранее в ходе данного TCP-сеанса. Описанная проверка предназначена для защиты сегментов некоторого TCP-сеанса от смешивания с сегментами других сеансов, но этот механизм защиты не так уж надежен, чем и пользуются злоумышленники.

Атака подделкой TCP-сегмента состоит в генерации TCP-сегментов, все атрибуты которых имеют значения, легитимные для некоторого существующего TCP-сеанса атакуемого компьютера, то есть IP-адреса, номера TCP-портов источника и приемника, а также порядковые номера из текущего диапазона окна приема. Принимающая сторона не может

отличить такие поддельные сегменты от настоящих и помещает информацию злоумышленника в поток пользовательских данных, а значит, злоумышленник может добиться желаемого эффекта, например, поместить ложную информацию в базу данных, заразить атакуемый компьютер вирусом и т. п.

Чтобы «поддельный» сегмент выглядел как настоящий, атакующий может либо прослушивать трафик, либо просто перебирать все возможные значения адресов, портов и порядковых номеров сегментов. Прослушивание трафика представляет собой отдельную нетривиальную задачу, включающую перенаправление трафика (об атаках такого типа см. далее). В то же время перебор параметров TCP-сеанса требует большой вычислительной мощности компьютера атакующего. В обоих случаях атаковать проще длительные TCP-сеансы, например сеансы загрузки больших видеофайлов (короткие сеансы веб-серфинга намного менее уязвимы).

Разновидностью подделки TCP-сегментов является их *повторное использование*. Если злоумышленник смог каким-то образом перехватить трафик между двумя участниками TCP-сеанса, то впоследствии он может просто посылать участникам сеанса дубликаты перехваченных сегментов. Этот прием может применяться злоумышленником для разных целей — например, для нарушения работы некоторого приложения за счет представления устаревшей (перехваченной) информации в качестве новой.

Сброс TCP-соединения

Атака сбросом TCP-соединения используется для разрыва TCP-соединений легальных пользователей. Для проведения атаки злоумышленник должен подделать заголовок TCP-сегмента. При поступлении TCP-сегмента с установленным флагом *RST* узел должен немедленно завершить сеанс, к которому относится этот сегмент, и удалить все данные, полученные в ходе сеанса. Разработчики протокола TCP ввели этот флаг для обработки аварийных ситуаций. Например, если в одном из узлов во время TCP-сеанса происходит сбой, то после восстановления системы он может послать сегмент с этим признаком, чтобы уведомить узел-собеседник о невозможности продолжения сеанса. Сброс соединения используется также некоторыми файерволами для прекращения атаки.

Борьба с атаками подделкой TCP-сегмента и сбросом TCP-соединения может вестись по двум направлениям. Первое направление связано с предотвращением прослушивания трафика. Второе направление основано на изменении поведения самого протокола TCP — например, путем включения дополнительной процедуры аутентификации каждого TCP-сегмента с использованием цифровой подписи. Как известно, цифровая подпись не обеспечивает конфиденциальности (содержимое защищаемых полей не шифруется), но она гарантирует, что TCP-сегмент не был изменен третьей стороной.

ICMP-атаки

Атака перенаправлением трафика может быть осуществлена в самых разных целях (одна из них раскрыта в процессе рассмотрения атаки подделкой TCP-сегментов). При этом существует несколько способов перенаправления трафика. Так, в пределах локальной сети эту задачу можно решить *с помощью протокола ICMP*. В соответствии с данным протоколом маршрутизатор посылает хосту непосредственно присоединенной локальной сети ICMP-сообщение о перенаправлении маршрута при отказе этого маршрута или в тех

случаях, когда обнаруживает, что для некоторого адреса назначения хост использует нерациональный маршрут. На рис. 31.2 применяемый по умолчанию маршрутизатор R1, получив от хоста H1 пакет, адресованный хосту H2, определяет, что наилучший маршрут к хосту H2 пролегает через маршрутизатор R2. Маршрутизатор R1 отбрасывает полученный пакет и помещает его заголовок в ICMP-сообщение о перенаправлении маршрута, которое посылает хосту H1. В сообщении содержится IP-адрес альтернативного маршрутизатора R2, который теперь должен использовать хост, посылая данные хосту H2. Хост H1 вносит изменения в свою таблицу маршрутизации и с этого момента отправляет пакеты хосту H2 по новому скорректированному маршруту.

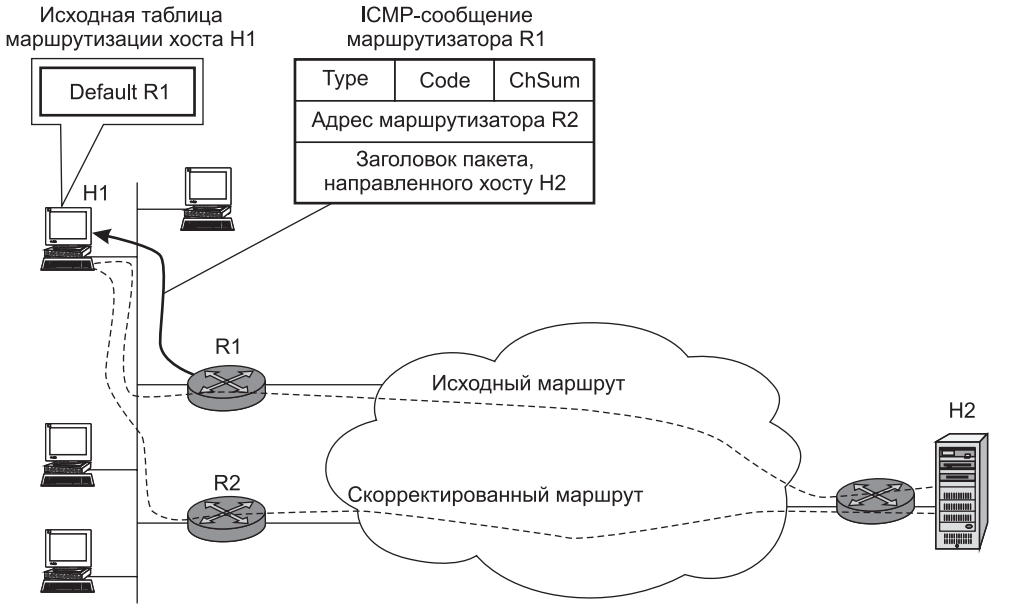


Рис. 31.2. Перенаправление маршрута предлагаемым по умолчанию маршрутизатором

Для перехвата трафика, направляемого хостом H1 хосту H2, злоумышленник должен сформировать и послать хосту H1 пакет, маскирующийся под ICMP-сообщение о перенаправлении маршрута (рис. 31.3). В этом сообщении содержится запрос о корректировке таблицы маршрутизации хоста H1, предусматривающей установку во всех пакетах с адресом IP_{H2} в качестве адреса следующего маршрутизатора адреса IP_{HA}, являющегося де-факто адресом хоста-злоумышленника HA.

Чтобы хост поверил этому сообщению, в поле IP-адреса отправителя должен быть помещен адрес предлагаемого по умолчанию маршрутизатора R1. Когда пакеты, передаваемые введенным в заблуждение хостом, начнут поступать на узел злоумышленника, он может либо захватывать и не передавать эти пакеты дальше, имитируя для поддержания диалога приложение, которому эти пакеты предназначались, либо организовать транзитную передачу данных по указанному адресу назначения IP_{H2}. Читая трафик между узлами H1 и H2, злоумышленник получает всю необходимую информацию для несанкционированного доступа к серверу H2. Сами маршрутизаторы также могут реагировать на ICMP-сообщения

о перенаправлении маршрута, но обычно провайдеры отключают эту опцию для предотвращения атак данного типа.

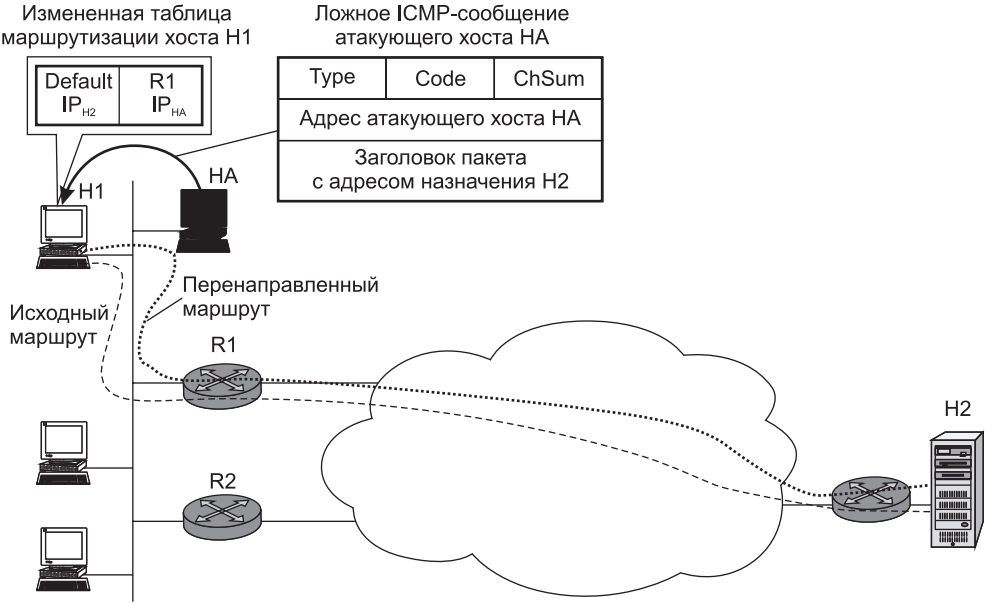


Рис. 31.3. Перенаправление маршрута злоумышленником

Заметим, что простейший вариант перенаправления трафика в локальной сети может быть осуществлен путем отправки в сеть *ложного ARP-ответа*. В данном случае схема очевидна: получив широковещательный ARP-запрос относительно некоторого IP-адреса, злоумышленник посылает ложный ARP-ответ, в котором сообщается, что данному IP-адресу соответствует его собственный MAC-адрес.

ICMP-атака Smurf — это DDoS-атака, использующая функцию *эхо-запроса* протокола ICMP. Название атаки произошло от имени файла smurf.c, содержащего код атаки и получившего распространение в 1998 году.

Эхо-запросы и эхо-ответы протокола ICMP больше известны по утилите *ping*[1], с помощью которой можно проверить достижимость узла интернета. Для проверки достижимости утилита ping посылает тестируемому узлу ICMP-пакет, в котором в качестве типа сообщения указан код 8 (эхо-запрос). Получив его, тестируемый узел отправляет в обратном направлении ICMP-пакет с кодом 0 (эхо-ответ). Атака Smurf тоже строится на возможности отправки эхо-запроса не только по индивидуальному, но и по *широковещательному* (broadcast) адресу некоторой сети. Например, если у сети адрес 200.200.100.0/24, то ее широковещательный адрес — 200.200.100.255, и эхо-запрос должен быть доставлен всем узлам этой сети (рис. 31.4).

[1] См. раздел «Сообщения эхо-запрос и эхо-ответ в утилите ping» в главе 14.

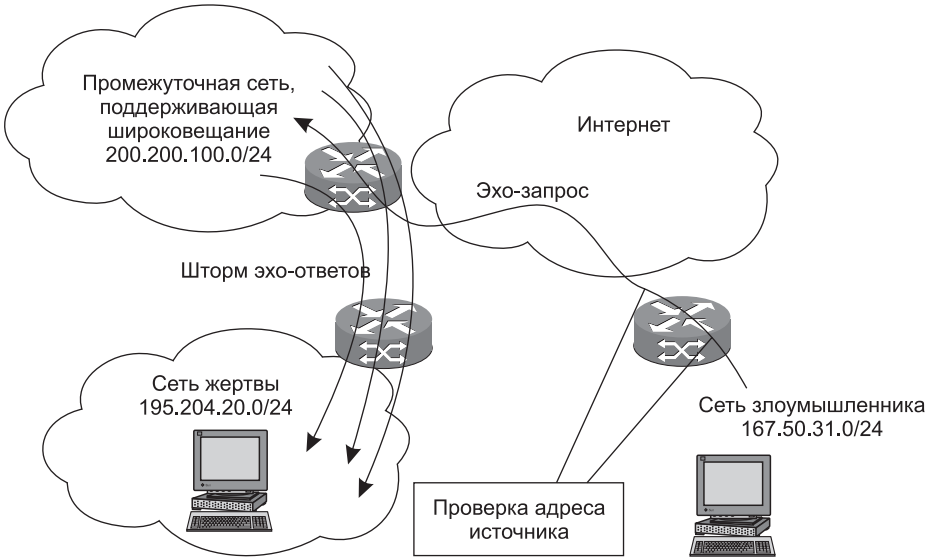


Рис. 31.4. Компоненты ICMP-атаки Smurf

Компьютер злоумышленника с адресом 167.50.31.17 находится в сети 167.50.31.0/24, а атакуемый компьютер имеет адрес 195.204.20.145 и подключен к сети 195.204.20.0/24. Компьютер злоумышленника генерирует эхо-запросы с адресом приемника 200.200.100.255 и адресом источника 195.204.20.145. Эхо-запросы передаются через интернет в сеть 200.200.100.0.24 и принимаются всеми узлами этой сети, которые отвечают на ICMP-запросы эхо-ответами. В том случае, когда в сети 200.200.100.255 имеется достаточно большое количество активных узлов (понятно, что их не может быть более 254), на атакуемый узел 195.204.20.145 приходит интенсивный поток эхо-ответов, так как именно его адрес указан в эхо-запросах как адрес источника. В результате сетевой интерфейс атакуемого компьютера оказывается затопленным эхо-ответами и при превышении некоторой величины интенсивности этого потока пропускная способность интерфейса оказывается исчерпанной.

В ICMP-атаке Smurf используется характерный прием – *усиление атаки за счет отражения* посланного пакета большим количеством компьютеров. Необходимо, впрочем, отметить, что сегодня ICMP-атака Smurf представляет скорее исторический интерес. До 1999 года передача через интернет IP-пакета с широковещательным адресом была обязательной для маршрутизаторов интернета, но из-за атак, подобных Smurf, в стандарты было внесено изменение, и сегодня предлагаемым по умолчанию режимом является *фильтрация пакетов с широковещательными адресами*. Кроме того, промежуточная сеть, узлы которой используются для отражения эхо-запроса, может быть экранирована с помощью файервола от эхо-запросов, приходящих из внешних сетей.

Атака с драматическим названием **«Пинг смерти»** (Ping of Death) состоит в отправке на атакуемый компьютер эхо-запроса в IP-пакете, длина которого превышает его допустимый размер, составляющий, согласно стандарту, 65 535 байт. Поскольку соответствующий буфер ядра ОС не рассчитан на такую длину пакета, ОС терпит крах, отсюда и название атаки, основанной на превышении размера буфера при сборке фрагментированного IP-пакета и, таким образом, являющейся частным случаем атак, использующих IP-фрагментацию

(см. далее). К слову, база для атаки «Пинг смерти» давно ликвидирована — разработчики ОС еще в середине 90-х годов ввели в стек TCP/IP проверку длины собираемого фрагментированного IP-пакета.

Другая атака, называемая **Ping-затоплением**, также является достаточно простой — злоумышленник использует утилиту ping своей ОС для отправки эхо-запросов на атакуемый компьютер с максимально возможной частотой. Если быстродействие сетевого интерфейса его компьютера выше, чем у атакуемого компьютера, то атака удается, так как вся входная пропускная способность интерфейса атакуемого компьютера оказывается исчерпанной. К тому же атакуемый компьютер будет успевать отвечать на часть эхо-запросов эхо-ответами, что приведет к частичному исчерпанию пропускной способности в выходном направлении, а также к замедлению работы программ из-за отвлечения центрального процессора на обработку эхо-запросов.

UDP-атаки

Атака UDP-затоплением относится к DoS-атакам и имеет целью исчерпание пропускной способности интерфейса атакуемого компьютера. Она подобна только что рассмотренной атаке Ping-затопления, когда злоумышленник просто направляет интенсивный поток UDP-дейтаграмм на атакуемый компьютер. Поскольку протокол UDP работает без установления соединения, то атакуемый компьютер обязан принимать все направляемые ему UDP-дейтаграммы, так как не может, как это делается при обмене данными по протоколу TCP, заставить передающий компьютер ограничить интенсивность потока направляемых ему пакетов, уменьшив размер окна приема. Злоумышленник может использовать аппаратный генератор трафика для того, чтобы генерировать UDP-трафик с максимально возможной скоростью выходного интерфейса, игнорируя ответные ICMP-сообщения в тех случаях, когда у атакуемого компьютера программный порт, указанный в UDP-пакетах, не открыт.

Слабым местом такого вида атак является то, что их интенсивность принципиально ограничена производительностью интерфейса атакующего компьютера. Имея стандартный для пользовательского компьютера интерфейс 1 Гбит/с, невозможно затопить UDP-пакетами сервер с интерфейсом 10 Гбит/с. Злоумышленник может преодолеть это ограничение, если в его распоряжении имеется сеть ботов. Именно такой подход был использован в 2007 году, когда была осуществлена массированная DDoS-атака UDP-затоплением на корневые DNS-серверы, при этом трафик создавался примерно пятью тысячами ботов (подробнее см. раздел «Атаки на DNS»).

Атака ICMP/UDP-затоплением имеет двойное имя, так как в ней используется два протокола. Злоумышленник направляет интенсивный поток UDP-пакетов, в которых в качестве адреса источника указан адрес компьютера-жертвы, на программные порты компьютеров, находящиеся в пассивном состоянии (то есть в данный момент с этими портами не связаны приложения, слушающие сеть). При получении UDP-пакета с номером пассивного порта компьютер в соответствии с логикой работы стека TCP/IP отвечает ICMP-сообщением о недостижимости порта назначения, которое направляется атакуемому компьютеру. Как видно из описания, в атаке имеет место отражение от компьютеров промежуточной сети; в случае использования широковещательного адреса она становится DDoS-атакой. Для предотвращения этой атаки применяют те же меры, что и для предотвращения ICMP-атаки Smurf, — дополнительно реализуется пропуск файерволом только тех UDP-пакетов, порты которых соответствуют активным приложениям компьютеров сети. Кроме того, можно

ограничить интенсивность сообщений о недостижимости порта назначения компьютеров сети.

Атака UDP/echo/chargen-затоплением похожа на описанную выше — в ней также имеет место отражение UDP-пакетов, но при этом пакеты отправляются с номером порта 7 или 19. Эти порты обычно активны, они поддерживают сервисы echo (порт 7) и chargen (порт 19), использующие протокол UDP. Сервис chargen в ответ на запрос генерирует строку случайных символов случайной длины от 0 до 512 и посылает ее обратившемуся хосту. Этот сервис был встроен в ОС Unix для отладки ее сетевых функций. Аналогичное назначение имеет сервис echo (не путать с эхо-запросами и эхо-ответами протокола ICMP), возвращающий строку любого запроса по адресу обратившегося хоста. В простейшем случае атакующий посылает UDP-пакеты на порт 7 и/или 19 некоторого промежуточного хоста и указывает обратный адрес атакуемого хоста. Промежуточный хост начинает «бомбардировать» атакуемый хост ответами сервисов chargen и/или echo. При этом усиления атаки не происходит, так как объем ответных сообщений невелик; для усиления может быть использован широковещательный адрес промежуточной сети. Более интересной выглядит атака, когда атакующий посылает пакет с портом 19 и указывает в нем исходный порт 7. В этом случае единственный пакет атакующего вызывает бесконечный обмен пакетами между сервисом chargen промежуточного хоста и сервисом echo атакуемого хоста.

IP-атаки

Протокол IP сам по себе не предоставляет злоумышленникам особых шансов для атак, так как работает без установления соединения и достаточно прост в реализации. Тем не менее некоторые возможности для IP-атак существуют. Рассмотрим их.

Атака на IP-опции представляет собой DoS-атаку на маршрутизаторы, в которой используется поле дополнительных опций протокола IP.

В *IPv4* заголовок IP-пакета может включать поле опций, задающих некоторую *нестандартную обработку* пакета маршрутизатором. Например, существует опция строгой маршрутизации от источника, позволяющая отправителю IP-пакета задать точный список адресов промежуточных маршрутизаторов, через которые должен проходить маршрут доставки пакета, в то время как опция свободной маршрутизации от источника задает лишь некоторые из промежуточных маршрутизаторов маршрута. Опция фиксации маршрута требует от маршрутизаторов фиксации в пакете адресов промежуточных маршрутизаторов, передающих пакет. Отметим, у производителей маршрутизаторов имеется возможность определять свои типы опций.

Поскольку у большинства IP-пакетов поле опций отсутствует, для продвижения пакетов маршрутизатор задействует специализированные *процессоры портов*, выполняющих эту операцию очень быстро и экономно. Но если встречается пакет с полем опций, то процессор порта его обработать не может и передает пакет *центральному процессору* маршрутизатора. В результате обработка трафика замедляется. Поэтому если на маршрутизатор поступает интенсивный поток пакетов, у которых присутствует одна или несколько опций, то его работа может существенно замедлиться, вплоть до отказа в обслуживании нормальных пакетов. Ситуация усугубляется, когда в пакете указаны две взаимоисключающие опции, например, строгой маршрутизации от источника и свободной маршрутизации от источника с разными промежуточными адресами.

Спецификация *IPv6* допускает наличие нескольких заголовков в пакете — основного и нескольких дополнительных. Вместо полей опций в пакете IPv6 могут присутствовать дополнительные заголовки, одним из которых является заголовок пошаговых опций (Hop-by-hop Options). Как и в случае опций IPv4, опции дополнительного заголовка пошаговых опций IPv6 обрабатываются центральным процессором маршрутизатора. Помещение в такой заголовок большого числа опций неопределенного типа будет замедлять работу маршрутизатора IPv6. Обычная практика борьбы с подобной атакой — фильтрация (отбрасывание) всех пакетов, в заголовке которых имеются опции. Возможно также игнорирование всех или некоторых опций.

Атака на фрагментацию направлена на конечные узлы IP-сетей, в обязанность которых входит сборка фрагментированного IP-пакета. Для подобных атак используются некоторые уязвимости, присущие операциям сборки, например:

- *Превышение максимальной длины пакета* (переполнение буфера сборки). Этот способ атаки уже был упомянут при описании атаки «Пинг смерти». Максимальное значение смещения фрагмента равно $(2^{13} - 1) \times 8 = 8191 \times 8 = 65\,528$. Так как максимальная длина IP-пакета равна 65 535 байт, очевидно, что последний фрагмент должен иметь длину не более 7 байт. Задавая фрагмент с максимальным смещением и размером в 8 байт и более, злоумышленник переполняет буфер ядра ОС, что может привести к падению ОС.
- *Перекрытие сегментов за счет специального подбора смещений и длин фрагментов.* Некоторые ОС не справляются со сборкой таких пакетов и падают. Например, эта уязвимость используется в атаке Teardrop.
- *Замещение фрагментов.* Эта DoS-атака используется для обмана таких защитных средств, как файерволы и системы обнаружения вторжений. Пакеты атаки фрагментируются и посылаются вместе с фрагментами-дубликатами, в которых содержится безобидная информация. Первым посылается безобидный фрагмент, а следом — фрагмент, содержащий код атаки, но с такими же смещением и длиной. В результате фрагмент атаки замещает безобидный фрагмент. Не все файерволы и системы обнаружения вторжений распознают фрагментированную таким образом атаку.

Сетевая разведка

Как можно увидеть из описания атак на транспортные протоколы, многие из них требуют предварительных знаний об атакуемой сети и ее хостах. Например, для проведения ICMP-атаки Smurf нужно найти промежуточную сеть с большим количеством хостов, отвечающих на эхо-запросы, при этом такая сеть должна быть достижима для пакетов с широковещательным адресом этой сети, посланных из сети злоумышленника; безусловно, должен быть известен и *IP-адрес* атакуемого компьютера. Если злоумышленник хочет задействовать сеть ботов, зараженных вирусом определенного типа, то ему понадобится просканировать большое количество компьютеров на отклик по определенному *порту*, который используется этим вирусом для получения команд от контроллера атаки. Дело в том, что вирусы стараются распространиться на возможно большее число компьютеров, но нельзя сказать заранее, будет ли успешным такое внедрение для какого-то определенного хоста — это зависит от конфигурации средств защиты и других параметров ОС хоста. Поэтому злоумышленник заранее не знает, какие хосты он может использовать в качестве членов сети ботов, даже если именно он инициировал распространение этого вируса. А возможно, он

просто решит воспользоваться известным вирусом, распространенным другими лицами, и поэтому ему нужно собрать сведения о зараженных компьютерах.

Вот почему почти любую атаку предваряет **сетевая разведка**, при которой злоумышленник пытается собрать необходимые для атаки сведения. Конкретный набор сведений зависит от типа атаки, но чаще всего сетевая разведка включает сбор следующих данных: IP-адреса активных (то есть включенных, отвечающих на сетевой трафик) хостов; номера активных TCP-портов; номера активных и пассивных UDP-портов хостов; тип и версии ОС и приложений.

Обнаружение IP-адресов активных хостов сети называют **сканированием сети** (network scanning), а активных и пассивных портов — **сканированием портов** (port scanning). Сам термин «сканирование» говорит о том, что злоумышленник тестирует один за другим все возможные значения IP-адресов некоторой подсети (например, для подсети с маской /24 — 254 значения) или номера портов (65 535 номеров и для TCP, и для UDP). Вот наиболее распространенные приемы сканирования сети:

- **Пинг**[1] **TCP SYN** к одному из публично доступных портов, чаще всего к порту 80 (порт веб-сервера), который с большой степенью вероятности (но, конечно, не обязательно) открыт для внешнего доступа. Если хост отвечает пакетом SYN/ACK, то сканер считает, что *хост активен*, и завершает TCP-соединение пакетом с признаком RST.
- **Пинг TCP ACK** позволяет во многих случаях обойти файервол, если тот блокирует выбранный порт. Обычной практикой конфигурирования файервола является разрешение трафика *уже установленных* TCP-соединений, а признаком принадлежности пакета к такому соединению является наличие установленного флага ACK. В том случае, когда пинг TCP SYN к некоторому порту не проходит, а TCP ACK проходит, результатом сканирования становится заключение: *хост активен, но защищен файерволом* — такая информация может быть ценной для злоумышленника.
- **Пинг UDP**. На тестируемый хост направляется UDP-пакет с номером порта, который, как рассчитывает злоумышленник, с большой степенью вероятности является *пассивным*. В том случае, когда компьютер включен и этот порт пассивен, сканер получает в ответ ICMP-сообщение о недоступности порта; если же компьютер отключен, то злоумышленник получает сообщение от маршрутизатора о недоступности хоста.
- **Пинг ICMP**. Администраторы сетей чаще всего блокируют эхо-запросы протокола ICMP, однако для проверки активности хоста злоумышленник может использовать *другие типы ICMP-запросов*, например запрос о длине маски IP-адреса (код 17) или запрос синхронизации времени протокола ICMP (код 13).
- **Пинг IP**. На исследуемый компьютер направляется IP-пакет с кодом протокола, отличным от кодов протоколов TCP, UDP и ICMP. Скорее всего, такой тип протокола не поддерживается стеком TCP/IP данного компьютера, и в том случае, если хост активен, в ответ будет послано ICMP-сообщение о недостижимости протокола.

Подобные же методы применяются и для *сканирования портов*. Здесь предпочтение отдается SYN-сканированию протокола TCP, так как это самый быстрый способ, что в данном случае имеет значение — в отличие от сканирования хостов здесь проверяются десятки ты-

---

[1] В этом перечне процедур термин «пинг» использован в широком смысле — он не указывает конкретно на утилиту ping, работающую по протоколу ICMP, а говорит о том, что данные процедуры, подобно утилите ping, тестируют активность хоста с определенным IP-адресом.

сяч портов (по 65 535 портов для TCP и UDP). Сканирование портов часто осуществляется с помощью тех же специализированных программных средств, что и для инвентаризации сети и аудита ее защищенности.

Сканирование сети и портов обычно не проходит незамеченным — очень вероятно, что средства протоколирования событий ОС и файерволов зафиксируют этот процесс, и администратор сканируемой сети начнет расследовать инцидент. И первый вопрос, возникающий при этом: с какого адреса выполнялось сканирование? Чтобы избежать раскрытия, злоумышленники часто используют *спуфинг IP-адреса*. На первый взгляд кажется, что в сетевой разведке этот прием не может сработать, так как злоумышленнику нужно получать ответы на свой компьютер, иначе он не может получать информацию. Тем не менее спуфинг IP-адреса возможен и при сканировании. Одним из приемов является маскировка его среди множества других адресов: тестовые сканирующие пакеты отправляются с действительного IP-адреса наряду с множеством таких же пакетов, но с поддельными адресами. Расчет здесь делается на то, что при расследовании факта сканирования трудно будет установить, кто являлся истинным организатором сканирования, а кого просто использовали как прикрытие. Еще более изощренным является так называемое «пустое» сканирование, когда истинный адрес никогда не указывается, а результаты сканирования злоумышленники пытаются понять по реакции третьего компьютера, адрес которого подделывается.

Атаки на DNS

Центральная роль службы DNS делает ее, с одной стороны, желанной *целью* атакующего, поскольку нарушение работы DNS наносит огромный ущерб работе сети, а с другой — мощным *средством* для проведения атак на другие сетевые механизмы, потому что многие из них оказываются безоружными перед этой глобальной службой.

DNS-спуфинг

В этой атаке DNS является не целью, а средством (рис. 31.5). Рассмотрим пример, в котором злоумышленник использует **DNS-спуфинг** для получения доступа к корпоративному серверу www.example.com. Для этого ему нужны аутентификационные данные какого-нибудь его клиента.

Он решает перенаправить поток данных, которые легальный корпоративный клиент посылает корпоративному серверу, на свой компьютер. Для этого нужно опередить ответ DNS-сервера резольверу клиента и навязать ему свой вариант ответа, в котором вместо IP-адреса корпоративного сервера (в примере — 193.25.34.125) злоумышленник указывает IP-адрес атакующего хоста (203.13.1.123). На пути реализации этого плана имеется несколько серьезных препятствий.

Прежде всего необходимо задержать ответ DNS-сервера, например, подвергнув его DoS-атаке. Другая проблема связана с определением номера порта DNS-клиента, который необходимо указать в заголовке пакета, чтобы данные дошли до приложения, так как если серверная часть DNS имеет постоянно закрепленный за ней так называемый хорошо известный номер порта 53, то клиентская часть протокола DNS получает номер порта динамически при запуске, причем ОС выбирает его из достаточно широкого диапазона. Эту задачу злоумышленник решает путем прямого перебора всех возможных номеров.



Рис. 31.5. Схема перенаправления трафика путем использования ложных DNS-ответов

Также путем перебора возможных значений преодолевается и проблема определения идентификаторов DNS-сообщений. Эти идентификаторы передаются в DNS-сообщениях и служат для того, чтобы DNS-клиент мог установить соответствие поступающих ответов посланным запросам. Итак, злоумышленник «бомбардирует» клиентскую машину ложными DNS-ответами, перебирая все возможные значения идентифицирующих полей так, чтобы клиент, в конце концов, принял один из них за истинный DNS-ответ. Как только это происходит, цель злоумышленника можно считать достигнутой: пакеты от клиента направляются на адрес атакующего хоста, злоумышленник получает в свое распоряжение имя и пароль легального пользователя, а с ними — и доступ к корпоративному серверу.

Атаки на корневые DNS-серверы

Наиболее мощными и ощутимыми по своим последствиям были DDoS-атаки на корневые серверы, случившиеся 21 октября 2002 года и 6–7 февраля 2007-го.

Подробности **атаки 21 октября 2002 года** приводятся в отчете специалистов, администрирующих корневые серверы[1]:

- Атака длилась чуть больше часа и была направлена на все 13 корневых серверов.
- Атака была комбинированной: использовались методы ICMP-атаки ping-затоплением, TCP-атаки затоплением SYN-пакетами, атаки фрагментированными IP-пакетами и атаки UDP-затоплением.

[1] https://www.caida.org/projects/dns/oct02dos/.

- Интенсивность атаки на сервер — 50–100 Мбит/с; суммарная интенсивность — 900 Мбит/с.
- В атаке использовался спуфинг IP-адресов, отследить реальные источники атаки не удалось.

Служба DNS показала хорошую устойчивость к атаке — пользователи замечали только небольшое увеличение времени ожидания при открытии сайта в браузере; все корневые серверы продолжали работать, и на все принятые ими запросы были даны ответы, но из-за перегрузки входных интерфейсов некоторых серверов не все запросы были приняты. После этой атаки были проведены дополнительные работы по повышению устойчивости службы DNS, которые включали повышение скорости интерфейсов и линий связи, соединяющих корневые серверы с интернетом, и увеличение числа корневых серверов. Кроме того, корневые серверы были более равномерно распределены по автономным системам и географическим регионам.

Атака 6–7 февраля 2007 года длилась 24 часа и была намного мощнее, чем атака 21 октября 2002 года, — интенсивность трафика достигала 1 Гбит/с на один пул корневых серверов, но атаковано было только четыре из них. В атаке было использовано 4500–5000 компьютеров под управлением Microsoft Windows, причем члены этой сети ботов были распределены по сетям нескольких стран. При атаке имело место затопление корневых серверов UDP-пакетами, направленными на порт 53 (порт DNS), то есть атака относилась к типу UDP-затопления, а использование порта 53 помогало пакетам добраться до серверов, так как у файерволов, защищающих DNS-серверы, этот порт всегда открыт, иначе сервер не смог бы выполнять свою работу. Атака привела к почти полному исчерпанию пропускной способности двух из четырех атакованных пулов серверов, но два других пострадали не так существенно и смогли отвечать на большую часть запросов.

Атака практически немедленно была обнаружена центрами, ответственными за администрирование атакованных пулов корневых серверов (по предупреждающим сообщениям самих серверов и данным хостов, выполняющих постоянный мониторинг корневых серверов путем отправки на них контрольных запросов). Для снижения эффекта атаки был предпринят ряд мер, первой из которых явилась блокировка любых DNS-запросов, длина которых превышала 300 байт (обычно длина DNS-запроса — не больше 100 байт), а в атакующих сообщениях для усиления эффекта затопления размеры полей данных UDP-потока доходили до 1023 байт. Однако такая блокировка помогла только частично — последующий анализ показал, что размер поля данных трафика атаки менялся случайным образом от 0 до 1023 байт, при этом атакующие компьютеры не использовали спуфинг IP-адресов, что дало возможность отследить размещение ботов: Южная Корея — 65 %, США — 19 %, Канада — 3,5 %, Китай — 2,5 %, остальные страны — 10 %. Хост, координирующий атаку, находился в США, хосты сети ботов обращались к нему по протоколу HTTP.

Причины атаки остались неясными; в отчете ICANN предполагается, что атака — проявление тщеславия хакеров: ведь попытка остановить весь интернет является вызовом для любого хакера.

Мы остановились на этих двух примерах, так как они дают хорошее представление о масштабах современных DDoS-атак и о том, что защититься от них очень сложно даже таким опытным специалистам, которые обслуживают корневые DNS-серверы. В то же время такое архитектурное решение, как виртуализация серверов, когда логический сервер представлен большим пулом физических серверов, рассредоточенных по разным сетям

и автономным системам, способно гасить эффект даже очень интенсивной DDoS-атаки. В эффективности такого подхода мы еще раз убедимся в главе 32 при рассмотрении безопасности облачных вычислений.

DDoS-атаки отражением от DNS-серверов

Основная идея **атаки отражением от DNS-серверов** базируется на следующем. В интернете работают миллионы DNS-серверов, отвечающих за отправку ответов на запросы клиентов. При этом ответ может по объему намного превосходить запрос — например, если запрос относится к передаче файла зоны (запрос типа AXFR) и зона включает большое количество записей. В марте 2013 года атаке такого рода[1] подвергся веб-сервер компании Spamhouse — некоммерческой организации, борющейся со спамом (рис. 31.6).

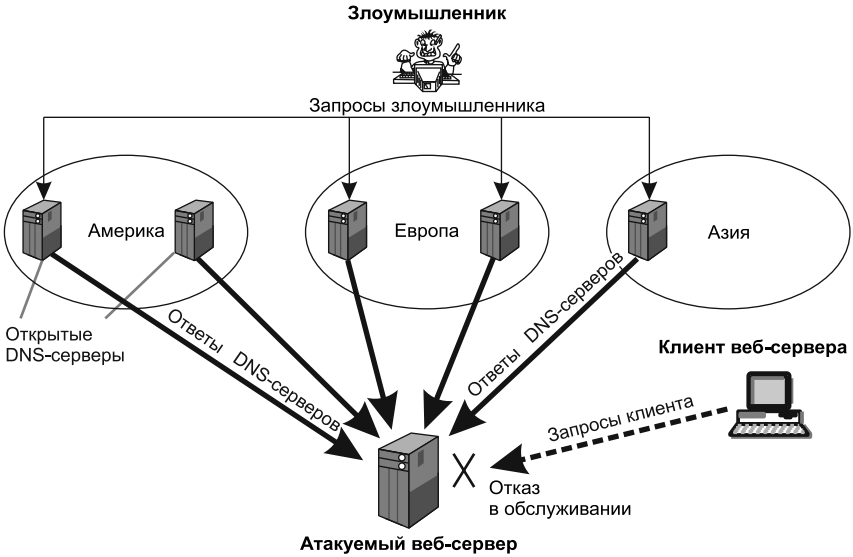


Рис. 31.6. Атака отражением от DNS- серверов

Для организации этой атаки было использовано около 30 000 DNS-серверов, работающих в открытом рекурсивном режиме, то есть отвечающих на запросы любых пользователей и при этом дающих полный (рекурсивный) ответ. Рекурсивный режим здесь является важным элементом атаки, так как нерекурсивные DNS-серверы только перенаправляют запрашивающего на другой DNS-сервер — их ответ является коротким и не может усилить атаку. Общей практикой является поддержание рекурсивного режима ответов только для своих клиентов — сотрудников предприятия для корпоративного DNS-сервера или же подписчиков сервиса для интернет-провайдера. Поскольку в интернете имеется около 28 миллионов открытых рекурсивных DNS-серверов, найти объекты для атаки оказалось не так уж трудно.

Злоумышленником был послан поток запросов на 30 000 открытых DNS-серверов, в качестве адреса отправителя которых указывался адрес атакуемого веб-сервера. Ответы от 30 000 DNS-серверов обрушились на веб-сервер компании Spamhouse.

[1] http://blog.cloudflare.com/the-ddos-that-knocked-spamhaus-offline-and-ho.

Для усиления атаки использовались не обычные запросы на разрешение имени, а запросы на передачу объемного файла со всеми записями зоны ripe.net (RIPE NCC — это региональный информационный интернет-центр по Европе). Файл зоны ripe.net имеет размер около 3000 байт, так что при размере запроса в 28 байт коэффициент усиления составил около 100. Такое мощное усиление позволило, используя поток запросов к одному DNS-серверу интенсивностью всего в 2,5 Мбит/с, создать атаку с общей интенсивностью в 75 Гбит/с. Для отдельного DNS-сервера такой поток запросов не является чем-то необычным, так что владельцы этих серверов, скорее всего, эту атаку не заметили, а вот результирующий поток атаки в 75 Гбит/с вывел веб-сервер компании Spamhouse из строя.

Точнее, веб-сервер Spamhouse был выведен из строя только до определенного момента, пока его владельцы не перевели его «под крыло» CloudFlare — провайдера облачных сервисов, к тому же специализирующегося на защите от DDoS-атак. Перевод помог, так как распределенная виртуальная структура CloudFlare, использующая технику anycast и файерволы, смогла абсорбировать большую часть трафика атаки, поле чего веб-сервер Spamhouse вновь стал доступен пользователям интернета.

Методы защиты службы DNS

Существует ряд мер предосторожности, которые повышают защищенность DNS-серверов от атак или использования их в качестве инструмента атаки:

- *Защита ОС хоста*. Так как DNS — это приложение ОС, то сама ОС должна быть надежно защищена всеми возможными способами.
- *Разделение пользователей на внутренних и внешних*. Рекурсивные неполномочные ответы должны предоставляться только внутренним пользователям как вызывающим большее доверие.
- *Передача файла зоны из первичного сервера только вторичным серверам этой зоны* с использованием для передачи защищенных протоколов, например SFTP или SCP.
- *Использование DNSSEC*. **DNSSEC** представляет собой набор стандартов, обеспечивающих аутентификацию ответов DNS-серверов с помощью цифровой подписи и системы публичных ключей. DNSSEC-клиент может проверить, что полученный ответ действительно пришел от полномочного сервера зоны, а не от сервера, который просто утверждает, что он полномочен, а на самом деле может таковым и не являться. DNSSEC затрудняет злоумышленникам спуфинг-атаки, так как для этого требуется подделывать цифровую подпись сервера. С 2010 года все корневые серверы, а также многие серверы верхнего уровня и крупных провайдеров поддерживают DNSSEC.

Безопасность маршрутизации на основе BGP

Уязвимости протокола BGP

Маршрутизация между автономными системами на основе протокола BGP является (наряду со службой DNS) одним из наиболее уязвимых элементов интернета. Это объясняется, во-первых, *тяжелыми последствиями*, к которым приводит ошибочная работа

BGP-маршрутизаторов сети провайдеров: маршруты ко многим частям интернета вдруг исчезают или оказываются ложными для значительной части пользователей. Во-вторых, протокол BGP принципиально *менее защищен*, чем внутренние протоколы маршрутизации OSPF и IS-IS, так как «собеседники» BGP-маршрутизатора находятся за пределами административной ответственности его организации, и поэтому возможностей для проверки достоверности маршрутных объявлений протокола BGP намного меньше, чем в случае внутренних протоколов.

Мы сознательно не использовали в заголовке этого раздела слово «атаки». Информация о случаях неправильной работы протокола BGP часто скрывается провайдерами интернета, избегающими раскрытия деталей работы своей сети по разным причинам, в том числе и по причине безопасности. Поэтому большая часть крупных инцидентов, произошедших в интернете по вине протокола BGP, остаются инцидентами, а не атаками, так как трудно сказать, произошел тот или иной инцидент из-за ошибки конфигурирования маршрутизатора персоналом провайдера или же это была спланированная и осуществленная атака. Во многих статьях и документах, описывающих уязвимости этого протокола маршрутизации, появилось новое действующее лицо — *провайдер-злоумышленник* (malicious ISP), который вольно или невольно создает проблемы для остальных провайдеров.

Инциденты с маршрутизацией в интернете являются следствием того, что маршрутное объявление протокола BGP формируется шаг за шагом многими провайдерами, при этом достоверность информации каждого шага проверить невозможно, так что у провайдера имеется полная свобода действий при обработке маршрутного объявления и передаче его соседним провайдерам. Вместо корректного объявления о префиксе своей сети с указанием номера своей автономной системы как исходной или добавления номера своей AS к уже имеющейся последовательности AS-номеров при трансляции объявления о чужой сети, он может выполнить ряд некорректных манипуляций с маршрутным объявлением, например:

- Поместить адрес чужой сети с номером своей автономной системы в качестве исходной, чтобы ложно направить трафик к этой сети в свою автономную систему. При этом адрес чужой сети в некорректном объявлении BGP должен быть *более специфическим*, чем уже существующие корректные записи об этой сети в маршрутизаторах других провайдеров, тогда новая ложная запись станет более приоритетной и будет использоваться вместо корректных. Такой тип инцидента называется *захватом префикса*.
- Выбросить из последовательности какую-то определенную автономную систему, чтобы обойти политику некоторой третьей автономной системы, которая по финансовым или иным соображениям блокирует все маршруты, проходящие через эту автономную систему.
- Добавить номер соседней автономной системы перед передачей ее объявления, чтобы соседняя автономная система, получив объявление и увидев в нем свой номер, отбросила его, решив, что объявление зациклилось.
- Добавить номер своей автономной системы несколько раз, чтобы объявление стало непривлекательным для других провайдеров (из-за размера последовательности AS).
- Составить ложную последовательность автономных систем, но поместить в качестве исходного правильный (но не свой) номер AS, чтобы вызвать доверие к маршруту.

Как видим, незащищенность маршрутного объявления дает большой простор для злонамеренных искажений и просто ошибок при его обработке. Вероятность ошибки усугубляется тем, что в отличие от внутренних протоколов маршрутизации, которые обрабатывают со-

общения с минимальным вмешательством администратора, работа протокола BGP обычно регулируется большим количеством правил фильтрации, которые задаются вручную администратором AS. Эти фильтры определяют *политику маршрутизации* той или иной автономной системы, отражающую взаимоотношения данного провайдера с каждым из провайдеров, с которым у него есть пиринговые соглашения о передаче трафика. Вместе с тем при правильном применении фильтры политики BGP являются мощным инструментом защиты BGP-маршрутизации от ошибок и атак.

Инциденты с протоколом BGP

Первый широко известный масштабный инцидент с BGP-маршрутизацией ярко выявил уязвимости BGP, которые затем много раз проявляли себя в аналогичных ситуациях. Этот инцидент произошел 25 апреля 1997 года, когда немало провайдеров обнаружило, что в их маршрутизаторах исчезли маршруты, описывающие путь ко многим сетям интернета. Причина этого неприятного обстоятельства довольно быстро была найдена: в объявлениях автономной системы **AS7007** указывалось, что путь к исчезнувшим сетям интернета должен пролегать через ее сети, хотя на самом деле эта небольшая автономная система не являлась транзитной для этих маршрутов. Звонок провайдеру AS7007 помог устранить причину: оказалось, что виновником был единственный маршрутизатор, который после реконфигурирования начал генерировать некорректные маршрутные объявления. Некорректность состояла в том, что в объявлениях указывались адреса не собственных сетей AS7007, а адреса сетей, принадлежащих другим провайдерам интернета. Кроме того, эти адреса оказались *более специфическими*, чем корректные адреса этих же сетей в маршрутизаторах провайдеров интернета, поэтому весь трафик к этим сетям стал направляться в AS7007 и там теряться, так как всех этих сетей у названного провайдера не было. После отключения виновного маршрутизатора таблицы маршрутизации провайдеров быстро восстановились, однако шок от того, как просто оказалось вывести интернет из строя, остался надолго.

Инцидент с AS7007 был первой масштабной демонстрацией уязвимости маршрутизации на основе протокола BGP — протокола, который, как и другие протоколы стека TCP/IP, был разработан в расчете на добрую волю всех пользователей интернета и не имел никакой защиты от ошибок или злого умысла. В дальнейшем подобные инциденты повторялись достаточно регулярно, причем один из них привлек большое внимание, потому что привел к временной недоступности популярного сервиса Youtube — в 2008 году оператор Pakistan Telecom пытался заблокировать доступ к Youtube для пользователей Пакистана, а вместо этого распространил всем провайдерам интернета объявления о том, что специфические маршруты к Youtube ведут через его сеть, что привело к направлению мирового трафика Youtube в сеть Pakistan Telecom в течение двух часов.

В представленном на рис. 31.7 примере диапазон адресов 12.24.0.0/16 принадлежит AS6. BGP-маршрутизатор этой автономной системы объявляет о достижимости данных адресов через свою сеть. Маршрутизаторы автономной системы AS5 принимают это объявление, помещают запись о достижимости адресов 12.24.0.0/16 в свои таблицы маршрутизации и передают его далее маршрутизаторам автономных систем AS4 и AS7. Теперь посмотрим, что произойдет, если маршрутизаторы автономной системы AS1 начнут по ошибке или умышленно распространять маршрутные объявления к адресам диапазона 12.24.128.0/17.

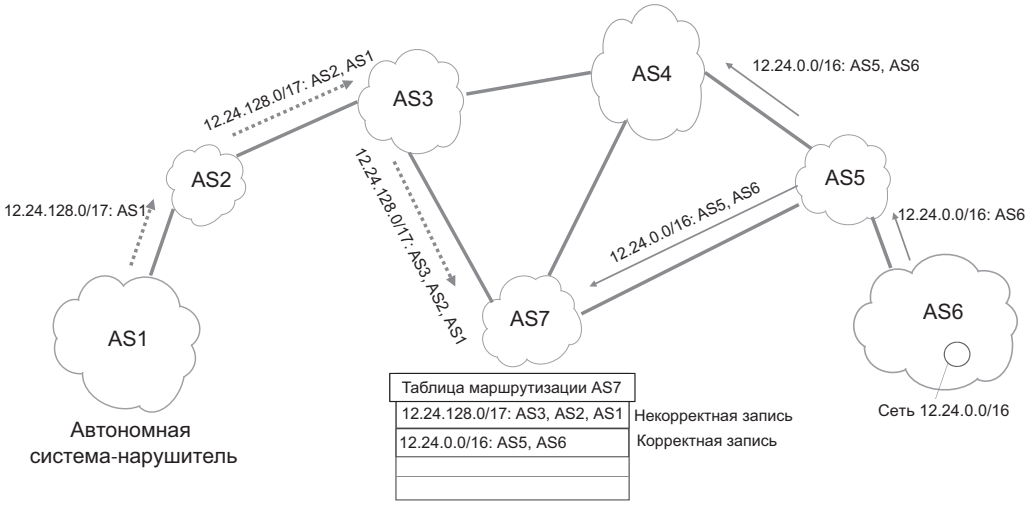


Рис. 31.7. Эффект захвата префикса адресов

Этот диапазон является поддиапазоном диапазона 12.24.0.0/16, но его префикс длиннее, то есть этот адрес является *более специфическим адресом*, чем адрес 12.24.0.0/16, поэтому маршрутизатор должен отдавать ему предпочтение перед более коротким адресом. В результате маршрутизаторы AS7 помещают запись о том, что трафик к хостам с адресами из диапазона 12.24.128.0/17 должен направляться в AS1, а не AS6. Эта информация распространится далее по всем автономным системам интернета, в том числе по AS6, что приведет к потере связи с хостами 12.24.128.0/17, так как они находятся в AS6, а не в AS1. Если система AS1 будет распространять только объявление 12.24.128.0/17, то остальные хосты из диапазона 12.24.0.0/16 окажутся достижимыми в AS6, если же AS1 будет распространять подобные объявления для всех поддиапазонов этого диапазона с маской 17, такие как 12.24.0.0/17, 12.24.192.0/17, 12.24.224.0/17 и т. д., то все хосты сети 12.24.0.0/16 станут недостижимыми.

(S) *Защита BGP*

Технологии защищенного канала

Известно, что задачу защиты данных можно разделить на две подзадачи: защиту данных внутри компьютера и защиту данных в процессе их передачи от одного компьютера к другому. Для обеспечения безопасности данных при их передаче по публичным сетям используются различные технологии защищенного канала.

Технология защищенного канала обеспечивает защиту трафика между двумя точками в открытой транспортной сети, например в интернете. Защищенный канал подразумевает выполнение трех основных функций:

- взаимная аутентификация абонентов при установлении соединения, которая может быть выполнена, например, путем обмена паролями;

- защита передаваемых по каналу сообщений от несанкционированного доступа, например, путем шифрования;
- подтверждение целостности поступающих по каналу сообщений, например, путем передачи одновременно с сообщением его дайджеста.

Способы образования защищенного канала

В зависимости от месторасположения программного обеспечения защищенного канала различают две схемы его образования:

- схема с конечными узлами, взаимодействующими через публичную сеть (рис. 31.8, *а*);
- схема с оборудованием поставщика услуг публичной сети, расположенным на границе между частной и публичной сетями (рис. 31.8, *б*).

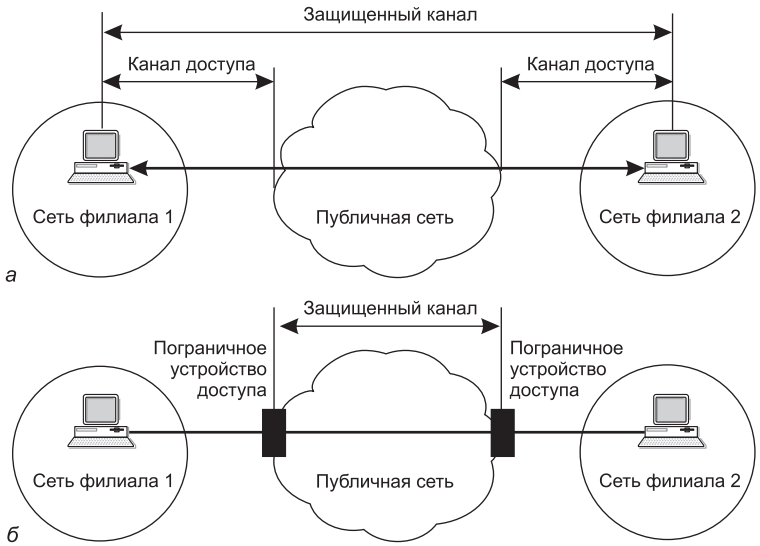


Рис. 31.8. Два подхода к образованию защищенного канала

В первом случае защищенный канал образуется программными средствами, установленными на двух удаленных компьютерах, принадлежащих двум разным локальным сетям одного предприятия и связанных между собой через публичную сеть. Преимуществом этого подхода является полная защищенность канала вдоль всего пути следования, а также возможность использования любых протоколов создания защищенных каналов, лишь бы на конечных точках канала поддерживался один и тот же протокол. Недостатки заключаются в избыточности и децентрализованности решения. Избыточность состоит в том, что вряд ли стоит создавать защищенный канал на всем пути следования данных: уязвимыми для злоумышленников обычно являются сети с коммутацией пакетов, а не каналы телефонной сети или выделенные каналы, через которые локальные сети подключены к территориальной сети. Поэтому защиту каналов доступа к публичной сети можно считать избыточной. Децентрализация заключается в том, что для каждого компьютера,

которому требуется предоставить услуги защищенного канала, необходимо отдельно устанавливать, конфигурировать и администрировать программные средства защиты данных. Подключение каждого нового компьютера к защищенному каналу требует выполнять эти трудоемкие операции заново.

Во втором случае клиенты и серверы не участвуют в создании защищенного канала — он прокладывается только внутри публичной сети с коммутацией пакетов, например, внутри интернета. Так, канал может быть проложен между сервером удаленного доступа поставщика услуг публичной сети и пограничным маршрутизатором корпоративной сети. Это хорошо масштабируемое решение, управляемое централизованно администраторами как корпоративной сети, так и сети поставщика услуг. Для компьютеров корпоративной сети канал прозрачен — программное обеспечение этих конечных узлов остается без изменений. Такой гибкий подход позволяет легко образовывать новые каналы защищенного взаимодействия между компьютерами независимо от места их расположения. Реализация этого подхода сложнее, поскольку требуется и стандартный протокол образования защищенного канала, и установка у всех поставщиков услуг программного обеспечения, поддерживающего такой протокол, и поддержка протокола производителями пограничного коммуникационного оборудования. Кроме того, вариант, когда все заботы по поддержанию защищенного канала берет на себя поставщик услуг публичной сети, оставляет сомнения в надежности защиты: каналы доступа к публичной сети оказываются незащищенными, а потребитель услуг чувствует себя в полной зависимости от надежности их поставщика.

Иерархия технологий защищенного канала

Защищенный канал можно построить с помощью системных средств, реализованных на разных уровнях модели OSI (рис. 31.9).



Рис. 29.9. Протоколы, формирующие защищенный канал на разных уровнях модели OSI

Если защита данных осуществляется средствами верхних уровней (прикладного, представления или сеансового), то такой способ защиты не зависит от технологий транспортировки

данных (IP или IPX, Ethernet или MPLS), что можно считать несомненным достоинством. В то же время приложения при этом становятся зависимыми от конкретного протокола защищенного канала, так как в них должны быть встроены явные вызовы функций этого протокола.

Защищенный канал, реализованный на самом высоком (*прикладном*) уровне, защищает только вполне определенную сетевую службу, например файловую, гипертекстовую или почтовую. Так, протокол **S/MIME** защищает исключительно сообщения электронной почты. При таком подходе для каждой службы необходимо разрабатывать собственную защищенную версию протокола.

Работа протокола защищенного канала на уровне *представления* делает его более универсальным средством, чем протокол безопасности прикладного уровня. Однако для того чтобы приложение смогло воспользоваться протоколом уровня представления, в него попрежнему приходится вносить исправления, хотя и не столь существенные, как в случае протокола прикладного уровня. Модификация приложения в данном случае сводится к встраиванию явных обращений к API соответствующего протокола безопасности.

На уровне представления работает протокол безопасности транспортного уровня **TLS** (Transport Layer Security), заменивший протокол защищенных сокетов **SSL** (Secure Socket Layer), который в настоящее время признан уязвимым и не рекомендуется для использования. Протокол SSL был разработан компанией Netscape Communications для защиты данных, передаваемых между веб-сервером и веб-браузером, но он мог быть использован и любыми другими приложениями. Для установления защищенного канала оба протокола используют следующие технологии безопасности:

- взаимная аутентификация приложений на обоих концах защищенного канала выполняется путем обмена сертификатами (стандарт X.509);
- для контроля целостности передаваемых данных используются дайджесты;
- секретность обеспечивается шифрованием с использованием симметричных ключей сеанса.

Средства защищенного канала становятся прозрачными для приложений в тех случаях, когда безопасность обеспечивается на *сетевом* и *канальном* уровнях. Однако здесь мы сталкиваемся с другой проблемой — зависимостью сервиса защищенного канала от протокола нижнего уровня. Например, протокол PPTP, не являясь протоколом канального уровня, защищает кадры протокола PPP канального уровня, упаковывая их в IP-пакеты. При этом не имеет никакого значения, пакет какого протокола, в свою очередь, упакован в данном PPP-кадре: IP, IPX, SNA или NetBIOS. С одной стороны, это делает сервис PPTP достаточно универсальным, так как клиент сервиса защищенного канала может задействовать любые протоколы в своей сети. С другой стороны, такая схема предъявляет жесткие требования к типу протокола канального уровня, используемому на участке доступа клиента к защищенному каналу, — для протокола PPTP таким протоколом может быть *только PPP*. Хотя протокол PPP очень распространен в линиях доступа, сегодня конкуренцию ему составляют протоколы Gigabit Ethernet и Fast Ethernet, которые все чаще работают не только в локальных, но и в глобальных сетях.

Работающий на сетевом уровне протокол IPSec является компромиссным вариантом. С одной стороны, он прозрачен для приложений, с другой — может работать практически во всех сетях, так как основан на широко распространенном протоколе IP и использует любую технологию канального уровня (PPP, Ethernet, MPLS и т. д.).

Система IPSec

Протокол IPSec в стандартах интернета называют *системой*. Действительно, IPSec — это согласованный набор открытых стандартов, имеющий сегодня вполне очерченное ядро, которое в то же время может быть достаточно просто дополнено новыми функциями и протоколами.

Ядро IPSec составляют три протокола:

- AH (Authentication Header — заголовок аутентификации), который гарантирует целостность и аутентичность данных;
- ESP (Encapsulating Security Payload — инкапсуляция зашифрованных данных), который шифрует передаваемые данные, обеспечивая конфиденциальность, может также поддерживать аутентификацию и целостность данных;
- IKE (Internet Key Exchange — обмен ключами интернета), который решает вспомогательную задачу автоматического предоставления конечным точкам защищенного канала секретных ключей, необходимых для работы протоколов аутентификации и шифрования данных.

Как видно из краткого описания функций, возможности протоколов AH и ESP частично перекрываются (рис. 29.10). В то время как AH отвечает только за обеспечение целостности и аутентификации данных, ESP может шифровать данные и, кроме того, выполнять функции протокола AH (хотя, как увидим позднее, аутентификация и целостность обеспечиваются им в несколько урезанном виде). ESP может поддерживать функции шифрования и аутентификации/целостности в любых комбинациях, то есть либо всю совокупность функций, либо только аутентификацию/целостность, либо только шифрование.

<table>
<tr><th>Выполняемые функции</th><th colspan="2">Протокол</th></tr>
<tr><td>Обеспечение целостности</td><td rowspan="2">AH</td><td rowspan="3">ESP</td></tr>
<tr><td>Обеспечение аутентичности</td></tr>
<tr><td>Обеспечение конфиденциальности (шифрование)</td><td></td></tr>
<tr><td>Распределение секретных ключей</td><td colspan="2">IKE</td></tr>
</table>

Рис. 31.10. Распределение функций между протоколами IPSec

Частичное дублирование функций защиты протоколами AH и ESP связано с применяемой во многих странах практикой ограничения экспорта и/или импорта средств, обеспечивающих конфиденциальность данных путем шифрования. Каждый из этих протоколов может использоваться как самостоятельно, так и одновременно с другим, так что в тех случаях, когда шифрование из-за действующих ограничений применять нельзя, систему можно поставлять только с протоколом AH. Естественно, подобная защита данных во многих случаях оказывается недостаточной. Принимающая сторона получает лишь возможность проверить, что данные были отправлены именно тем узлом, от которого они ожидаются, и дошли в том виде, в котором были отправлены. Однако от несанкционированного просмотра данных на пути их следования по сети протокол AH защитить не может, так как не шифрует их, — для шифрования данных необходим протокол ESP.

Безопасная ассоциация

Чтобы протоколы AH и ESP могли выполнять свою работу по защите передаваемых данных, протокол IKE устанавливает между двумя конечными точками *логическое соединение*, которое в стандартах IPSec носит название **безопасной ассоциации** (Security Association, **SA**).

Стандарты IPSec позволяют конечным точкам защищенного канала использовать единственную безопасную ассоциацию для передачи трафика всех взаимодействующих через этот канал хостов или создавать для этой цели произвольное число безопасных ассоциаций, например, по одной на каждое TCP-соединение. Это дает возможность выбирать нужную степень детализации защиты — от одной общей ассоциации для трафика множества конечных узлов до индивидуально настроенных ассоциаций для защиты каждого приложения.

Безопасная ассоциация в протоколе IPSec представляет собой *однонаправленное* (симплексное) логическое соединение, поэтому если требуется обеспечить безопасный двусторонний обмен данными, то необходимо установить две безопасные ассоциации. Эти ассоциации в общем случае могут иметь разные характеристики, например, при передаче запросов к базе данных достаточно только аутентификации, а для ответных данных, несущих ценную информацию, дополнительно может потребоваться обеспечить и их конфиденциальность.

Установление безопасной ассоциации начинается с взаимной *аутентификации* сторон, потому что все меры безопасности теряют смысл, если данные передаются или принимаются не тем лицом или не от того лица. Выбираемые далее параметры SA определяют, какой из двух протоколов, AH или ESP, будет применяться для защиты данных, какие функции будет выполнять протокол (например, можно выполнять только аутентификацию и проверку целостности, а можно, кроме того, еще и обеспечивать конфиденциальность). Очень важными параметрами безопасной ассоциации являются также *секретные ключи*, используемые в работе протоколов AH и ESP.

Протокол IPSec допускает как *автоматическое*, так и *ручное* установление безопасной ассоциации. При ручном способе администратор конфигурирует конечные узлы так, чтобы они поддерживали согласованные параметры ассоциации, включая секретные ключи. При автоматической процедуре установления SA протоколы IKE, работающие по разные стороны канала, выбирают параметры в ходе переговорного процесса.

Для каждой задачи, решаемой протоколами AH и ESP, предлагается несколько схем аутентификации и шифрования (рис. 31.11). Это делает протокол IPSec очень гибким средством. Заметим, что выбор дайджест-функции для решения задач целостности и аутентификации никак не влияет на выбор функции шифрования, обеспечивающей конфиденциальность данных.

Для обеспечения совместимости в стандартной версии IPSec определен некоторый обязательный «инструментальный» набор, в частности, для аутентификации данных всегда может быть использована одна из стандартных дайджест-функций MD5 либо SHA-1, а в число алгоритмов шифрования непременно входит DES. При этом производители продуктов, в которых используется IPSec, вольны расширять протокол путем включения других алгоритмов аутентификации и симметричного шифрования, что они с успехом и делают. Например, многие реализации IPSec поддерживают популярный алгоритм шифрования Triple DES, а также сравнительно новые алгоритмы: Blowfish, Cast, CDMF, Idea, RC5.

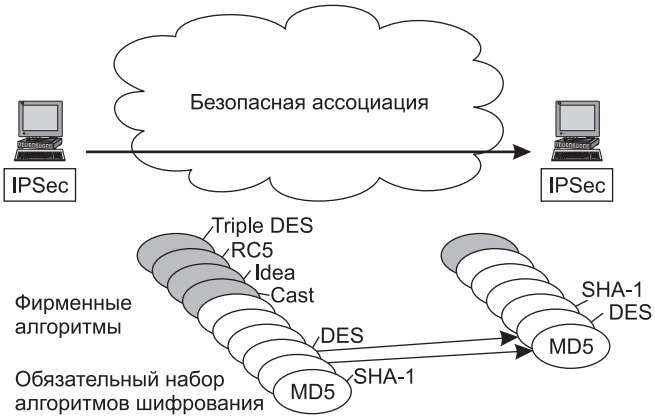


Рис. 31.11. Согласование параметров в протоколе ESP

Транспортный и туннельный режимы

Протоколы AH и ESP могут защищать данные в двух режимах: транспортном и туннельном. В **транспортном режиме** передача IP-пакета через сеть выполняется с помощью оригинального заголовка этого пакета, а в **туннельном режиме** исходный пакет помещается в новый IP-пакет, и передача данных по сети выполняется на основании заголовка нового IP-пакета.

Применение того или иного режима зависит от требований, предъявляемых к защите данных, а также от роли, которую играет в сети узел, завершающий защищенный канал. Так, узел может быть хостом (конечным узлом) или шлюзом (промежуточным узлом). Соответственно, возможны три схемы применения протокола IPSec:

- «хост-хост»;
- «шлюз-шлюз»;
- «хост-шлюз».

В схеме «хост-хост» безопасная ассоциация устанавливается между двумя конечными узлами сети. При этом протокол IPSec работает на каждом из этих узлов. Для схемы «хост-хост» чаще всего используется *транспортный* режим защиты.

В схеме «шлюз-шлюз» защищенный канал устанавливается между двумя промежуточными узлами, так называемыми **шлюзами безопасности** (Security Gateway, **SG**), на каждом из которых работает протокол IPSec (рис. 31.12). Защищенный обмен данными может происходить между любыми двумя конечными узлами, подключенными к сетям, которые расположены позади шлюзов безопасности. От конечных узлов поддержки протокола IPSec не требуется — они передают свой трафик в незащищенном виде через заслуживающие доверие внутренние сети предприятий. Трафик, направляемый в общедоступную сеть, проходит через шлюз безопасности, который и обеспечивает его защиту с помощью протокола IPSec. Шлюзам доступен только *туннельный* режим работы.

На рисунке пользователь компьютера с адресом IP1 посылает пакет по адресу IP2, используя туннельный режим протокола IPSec. Шлюз SG1 зашифровывает пакет вместе с заголовком, снабжая его новым IP-заголовком, в котором в качестве адреса отправите

ля указывает свой адрес — IP3, а в качестве адреса получателя — адрес IP4 шлюза SG2. Передача данных по составной IP-сети выполняется на основании заголовка внешнего пакета, а внутренний пакет становится при этом полем данных для внешнего пакета. На шлюзе SG2 протокол IPSec извлекает инкапсулированный пакет и расшифровывает его, приводя к исходному виду.

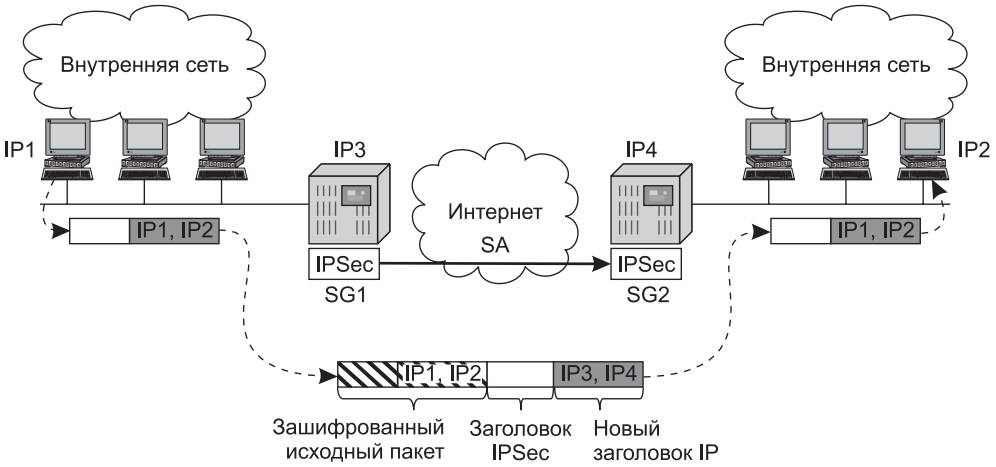


Рис. 31.12. Работа защищенного канала по схеме «шлюз-шлюз» в туннельном режиме

Схема «хост-шлюз» часто применяется при *удаленном доступе*. В этом случае защищенный канал прокладывается между удаленным хостом, на котором работает протокол IPSec, и шлюзом, защищающим трафик для всех хостов, входящих во внутреннюю сеть предприятия. Эту схему можно усложнить, создав параллельно еще один защищенный канал — между удаленным хостом и каким-либо хостом, принадлежащим внутренней сети, защищаемой шлюзом (рис. 31.13). Комбинированное использование двух безопасных ассоциаций позволяет надежно защитить трафик во внутренней сети.

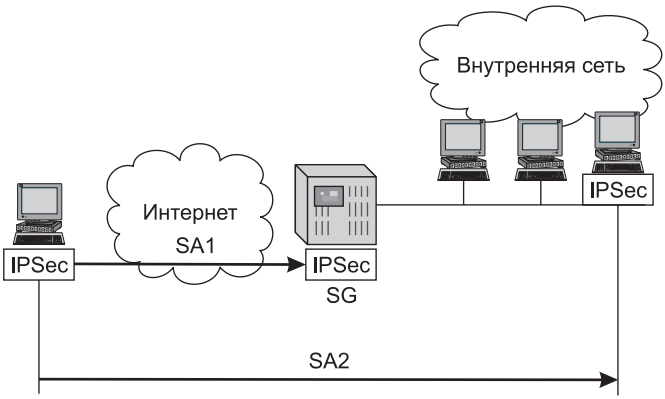


Рис. 31.13. Схема защищенного канала «хост-шлюз»

Протокол AH

Протокол AH позволяет приемной стороне убедиться, что:

- пакет был отправлен стороной, с которой установлена безопасная ассоциация;
- содержимое пакета не было искажено в процессе его передачи по сети;
- пакет не является дубликатом уже полученного пакета.

Две первые функции обязательны для протокола AH, а последняя выбирается по желанию при установлении ассоциации. Для выполнения этих функций протокол AH использует специальный заголовок (рис. 31.14).



Рис. 29.14. Структура заголовка протокола AH

В поле *следующего заголовка* (next header) указывается код протокола более высокого уровня, то есть протокола, сообщение которого размещено в поле данных IP-пакета. Скорее всего, им будет один из протоколов транспортного уровня (TCP или UDP) или протокол ICMP, но может встретиться и протокол ESP, если он используется в комбинации с AH.

В поле *длины полезной нагрузки* (payload length) содержится длина заголовка AH.

Индекс параметров безопасности (Security Parameters Index, SPI) служит для связи пакета с предусмотренной для него безопасной ассоциацией (подробнее см. ниже).

Поле *порядкового номера* (Sequence Number, SN) пакета применяется для защиты от его ложного воспроизведения (когда третья сторона пытается повторно использовать перехваченные защищенные пакеты, отправленные реально аутентифицированным отправителем). Отправляющая сторона последовательно увеличивает значение этого поля в каждом новом пакете, передаваемом в рамках данной ассоциации, так что приход дубликата обнаружится принимающей стороной (если, конечно, в рамках ассоциации будет активирована функция защиты от ложного воспроизведения). Однако в любом случае в функции протокола AH не входит восстановление утерянных и упорядочивание прибывающих пакетов — он просто отбрасывает пакет, когда обнаруживает, что аналогичный пакет уже получен. Чтобы сократить требуемую для работы протокола буферную память, используется механизм скользящего окна — на предмет дублирования проверяются только те пакеты, чей номер находится в пределах окна, обычно 32 или 64 пакета.

Поле *данных аутентификации* (authentication data) содержит хеш-код (дайджест) исходного IP-пакета, вычисленный с помощью одной из двух обязательно поддерживаемых

протоколом AH односторонних функций шифрования MD5 или SAH-1, но может использоваться и любая другая функция, о которой стороны договорились в ходе установления ассоциации. При вычислении дайджеста пакета в качестве параметра односторонней функции выступает симметричный секретный ключ, сгенерированный для данной ассоциации вручную или автоматически с помощью протокола IKE. Так как длина дайджеста зависит от выбранной функции, это поле имеет в общем случае переменный размер. Протокол AH старается охватить при вычислении дайджеста как можно большее число полей исходного IP-пакета, но некоторые из них в процессе передачи пакета по сети меняются непредсказуемым образом, поэтому не могут включаться в аутентифицируемую часть пакета. Например, целостность значения поля времени жизни (TTL) в приемной точке канала оценить нельзя, так как оно уменьшается на единицу каждым промежуточным маршрутизатором и никак не может совпадать с исходным.

Местоположение заголовка AH в пакете зависит от того, в каком режиме — транспортном или туннельном — сконфигурирован защищенный канал. Вид результирующего пакета в транспортном режиме представлен на рис. 31.15.

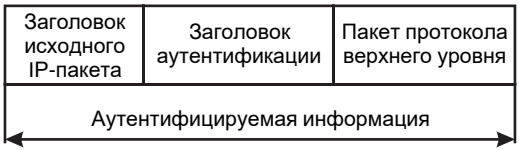


Рис. 31.15. Структура IP-пакета, обработанного протоколом AH в транспортном режиме

При использовании туннельного режима, когда шлюз IPSec принимает проходящий через него транзитом исходящий пакет и создает для него внешний IP-пакет, протокол AH защищает все поля исходного пакета, а также неизменяемые поля нового заголовка внешнего пакета (рис. 31.16).

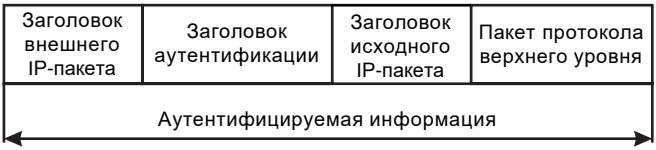


Рис. 31.16. Структура IP-пакета, обработанного протоколом AH в туннельном режиме

Протокол ESP

Протокол ESP решает две группы задач. К первой относятся задачи обеспечения аутентификации и целостности данных на основе дайджеста, аналогичные задачам протокола AH, ко второй — защита передаваемых данных путем их шифрования от несанкционированного просмотра. Как видно на рис. 31.17, заголовок делится на две части, разделяемые полем данных. Первая часть, называемая собственно *заголовком ESP*, образуется полями SPI и SN, назначение которых аналогично одноименным полям протокола AH, и размещается перед полем данных. Остальные служебные поля протокола ESP, называемые *концевиком ESP*, расположены в конце пакета.



Рис. 31.17. Структура IP-пакета, обработанного протоколом ESP в транспортном режиме

Два поля концевика — *следующего заголовка* и *данных аутентификации* — также аналогичны полям заголовка AH. Поле данных аутентификации отсутствует, если при установлении безопасной ассоциации принято решение не использовать средств протокола ESP, касающихся обеспечения целостности. Помимо этих полей концевик содержит два дополнительных поля — *заполнителя* и *длины заполнителя*. Заполнитель может понадобиться в трех случаях. Во-первых, для нормальной работы некоторых алгоритмов шифрования необходимо, чтобы шифруемый текст содержал кратное число блоков определенного размера. Во-вторых, формат заголовка ESP требует, чтобы поле данных заканчивалось на границе четырех байтов. И наконец, заполнитель можно использовать, чтобы скрыть действительный размер пакета в целях обеспечения так называемой частичной конфиденциальности трафика. Правда, возможность маскировки ограничивается сравнительно небольшим объемом заполнителя — 255 байт, поскольку большой объем избыточных данных может снизить полезную пропускную способность канала связи.

На рис. 31.17 показано размещение полей заголовка ESP в *транспортном режиме*. В этом режиме ESP не шифрует заголовок исходного IP-пакета, иначе маршрутизатор не сможет прочитать поля заголовка и корректно выполнить продвижение пакета между сетями. В число шифруемых полей не попадают также поля SPI и SN, которые должны передаваться в открытом виде, чтобы прибывший пакет можно было отнести к определенной ассоциации и предотвратить ложное воспроизведение пакета.

В *туннельном режиме* заголовок исходного IP-пакета помещается после заголовка ESP и полностью попадает в число защищаемых полей, а заголовок внешнего IP-пакета протоколом ESP не защищается (рис. 31.18).



Рис. 31.18. Структура IP-пакета, обработанного протоколом ESP в туннельном режиме

Базы данных SAD И SPD

При установлении безопасной ассоциации, как и при любом другом логическом соединении, две стороны принимают ряд соглашений, регламентирующих процесс передачи потока данных между ними и фиксируемых в виде набора параметров. Для безопасной ассоциации такими параметрами являются, в частности, тип и режим работы протокола защиты (AH или ESP), методы шифрования, секретные ключи, значение текущего номера пакета в ассоциации и другая существенная информация. Каким же образом протокол IPSec, работающий на хосте или шлюзе, определяет способ защиты, который он должен применить к трафику? Решение основано на использовании в каждом узле, поддерживающем IPSec, двух типов баз данных:

- *баз данных безопасных ассоциаций* (Security Associations Database, **SAD**), в которых хранятся наборы текущих параметров, определяющих все активные SA. Каждый узел IPSec поддерживает две базы SAD (для исходящих и входящих ассоциаций соответственно);
- *баз данных политики безопасности* (Security Policy Database, **SPD**), которые определяют соответствие между IP-пакетами и установленными для них правилами обработки. Записи SPD состоят из полей двух типов — полей селектора пакета и полей политики защиты для пакета с данным значением селектора.

Селектор в SPD (рис. 31.19) включает следующий набор признаков, на основании которых можно с большой степенью детализации выделить защищаемый поток:

- IP-адреса источника и приемника, которые могут быть представлены как в виде отдельных адресов (индивидуальных, групповых или широковещательных), так и диапазонами адресов, заданными с помощью верхней и нижней границ либо с помощью маски;
- порты источника и приемника (то есть TCP- или UDP-порты);
- тип протокола транспортного уровня (TCP, UDP);
- имя пользователя в формате DNS или X.500;
- имя системы (хоста, шлюза безопасности и т. п.) в формате DNS или X.500.

Для каждого нового пакета, поступающего в защищенный канал, IPSec просматривает все записи в базе SPD, сравнивая значение селекторов этих записей с соответствующими полями IP-пакета. Если значение полей совпадает с каким-либо селектором, то над пакетом выполняются действия, определенные в поле политики безопасности данной записи: передачу пакета без изменения, его отбрасывание либо обработку средствами IPSec. В последнем случае поле политики защиты должно содержать ссылку на запись в базе данных SAD, в которую помещен набор параметров безопасной ассоциации для данного пакета (на рисунке для исходящего пакета определена ассоциация SA3). На основании заданных параметров безопасной ассоциации к пакету применяются соответствующие протокол (на рисунке — ESP), функции шифрования и секретные ключи.

Если к исходящему пакету нужно применить некоторую политику защиты, но указатель записи SPD показывает, что в настоящее время нет активной безопасной ассоциации с требуемой политикой, то IPSec создает новую ассоциацию с помощью протокола IKE, помещая новые записи в базы данных SAD и SPD.

Базы данных политики безопасности создаются и администрируются либо пользователем (этот вариант больше подходит для хоста), либо системным администратором (вариант для шлюза), либо автоматически (приложением).

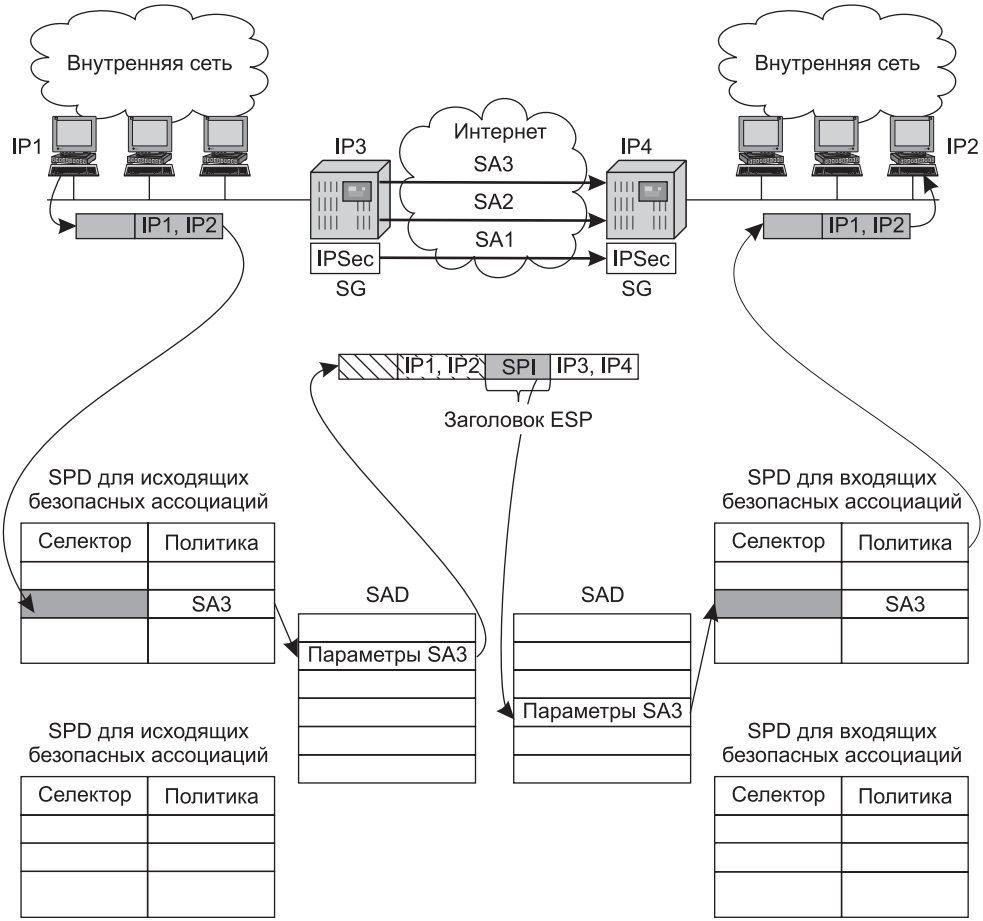


Рис. 31.19. Использование баз данных SPD и SAD

Ранее мы выяснили, что установление связи между исходящим IP-пакетом и заданной для него безопасной ассоциацией происходит путем селекции. Однако остается открытым другой вопрос: как *принимающий* узел IPSec определяет способ обработки прибывшего пакета, если при шифровании многие ключевые параметры пакета, отраженные в селекторе, оказываются недоступными, а значит, невозможно определить соответствующую запись в базах данных SAD и SPD и, следовательно, тип процедуры, которую надо применить к поступившему пакету? Для решения этой проблемы в заголовках AH и ESP и предусмотрено поле *SPI*. В это поле помещается указатель на ту строку базы данных SAD, в которой записаны параметры соответствующей безопасной ассоциации. Поле SPI заполняется протоколом AH или ESP во время обработки пакета в отправной точке защищенного канала. Когда пакет приходит в конечный узел защищенного канала, из его внешнего заголовка ESP или AH (на рисунке — из заголовка ESP) извлекается значение SPI, после чего обработка пакета выполняется с учетом всех параметров заданной этим указателем ассоциации.

Таким образом, для распознавания пакетов, относящихся к разным безопасным ассоциациям, используются:

- на узле-отправителе — селектор;
- на узле-получателе — индекс параметров безопасности (SPI) .

После дешифрования пакета приемный узел IPSec проверяет его признаки (ставшие теперь доступными) на предмет совпадения с селектором записи SPD для входящего трафика, чтобы убедиться, что ошибки не произошло и выполняемая обработка пакета соответствует политике защиты, заданной администратором.

Использование баз SPD и SAD для защиты трафика позволяет гибко сочетать механизм безопасных ассоциаций, который предусматривает установление логического соединения, с дейтаграммным характером трафика протокола IP.

(S) *VPN на основе IPSec*

ГЛАВА 32 Безопасность программного кода и сетевых служб

Уязвимости программного кода и вредоносные программы

Программная система, состоящая из десятков тысяч строк кода, всегда имеет уязвимости, которые может использовать злоумышленник. Эти уязвимости могут быть результатом ошибок программистов — в соответствии с исследованием CyLab Университета Карнеги Мэллона в среднем каждые 1000 строк кода содержат 20–30 ошибок, из которых 5 % влияют на безопасность системы, а 1 % открывает возможности для взлома системы.

Уязвимости, связанные с нарушением защиты оперативной памяти

Области оперативной памяти (адресные пространства) отдельных процессов защищены друг от друга. Защита памяти реализуется ОС в тесном взаимодействии с аппаратными механизмами процессора. Несмотря на это, некорректное использование областей памяти все же может происходить в пределах адресного пространства *отдельного процесса* или в области памяти *ядра* ОС. В последнем случае это особенно опасно, так как может вызвать крах всей системы, а не отдельного приложения, как в первом случае.

Переполнение буфера памяти является, по-видимому, наиболее часто используемой уязвимостью, связанной с нарушением защиты памяти. Мы уже знаем одну такую атаку, которая использует буфер, расположенный в памяти ядра, и приводит к краху всей системы — это атака «Пинг смерти». Точнее, приводила, так как ошибка в операционных системах, приводящая их к краху при превышении IP-пакетом размера в 65 535 байт, уже устранена. Тем не менее механизм, который эксплуатируется этой атакой, очень типичен — он использует отсутствие контроля над вводимой из внешнего мира информацией (в данном случае не контролируется длина помещаемого в буфер пакета).

Переполнение стека является частным случаем переполнения буфера памяти. Этот вид уязвимости часто используется злоумышленниками, чтобы заставить ОС выполнить код злоумышленника. Напомним, стеком является область памяти с реализацией стратегии записи LIFO (Last In First Out — последним пришел, первым вышел). Этот способ записи удобен при многократном вызове функций (подпрограмм), так как он обеспечивает экономичный возврат из вызванной функции в вызывающую.

Типичная структура стека, который растет в сторону меньших адресов (архитектура Intel x86), показана на рис. 30.1, *а*. Здесь мы видим стек, содержащий данные одной функции $f_1(A_1, A_2)$. В стек помещены аргументы этой функции, за которыми идет адрес возврата в функцию, ее вызвавшую — в данном случае это функция `main`, то есть основное тело программы, написанной на языке C. За адресом возврата идет локальная память функции f_1, используемая для хранения ее локальных переменных и массивов. Указатель стека содержит адрес первого слова свободной области стека.

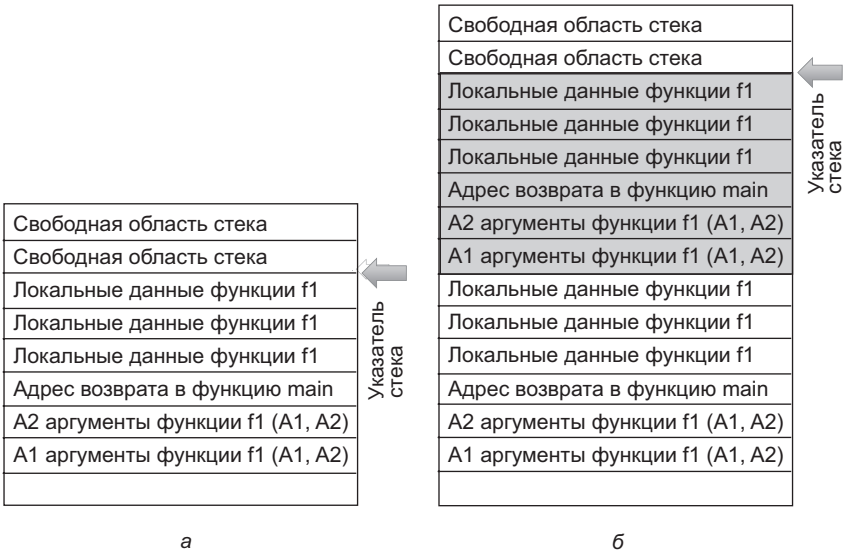


Рис. 30.1. *а* — структура стека до вызова функции f_2 из функции f_1; *б* — структура стека после вызова функции f_2 из функции f_1

Если функция f_1 вызывает другую функцию, f_2, то ее аргументы, адрес возврата в функцию f_1 и ее локальная память будут размещаться над областью памяти, выделенной в стеке функции f_1 (рис. 32.1, *б*). При завершении функция f_2 должна выполнить специальную инструкцию `RETURN`, которая вернет управление по адресу возврата в функцию f_1 и очистит стек от данных функции f_2, вернув указатель стека на прежнее место. Переполнение стека может произойти, если в область локальной памяти функции помещаются данные, длина которых больше длины этой области. В таком случае эти данные могут наложиться на адрес возврата. В результате после завершения вызванной функции произойдет переход на некоторый адрес, который может быть как случайным, так и специально сформирован злоумышленником. Многие атаки основаны на том, что в область локальных данных стека помещается вредоносный код, которому затем передается управление за счет подмены содержимого поля адреса возврата адресом начала вредоносного кода.

Некоторые операционные системы помечают область стека как неисполняемую, что предотвращает выполнение вредоносного кода в случае его попадания в стек.

(S) *Пример использования техники переполнения стека для организации атаки*

(S) *Скрытые коммуникации и скрытые каналы*

Переполнение буфера является частным случаем уязвимостей, являющихся следствием слабого контроля вводимых данных. В более общем случае — *любая* непредвиденная создателем программы форма вводимых данных может вызвать совершено неожиданные последствия, и этот факт может быть использован злоумышленником. Как любят повторять специалисты по разработке безопасного кода, «любой ввод данных — это зло».

Тривиальным примером является веб-форма, в которой пользователю предлагается ввести номер статьи, выбранный из списка, включающего 10 статей. Если разработчик не предвидел, что вместо ожидаемого положительного числа из диапазона от 1 до 10 пользователь может ввести отрицательное, к примеру, –1, то приложение может повести себя совсем не так, как он планировал, например, выдать конфиденциальный документ вместо публично доступной статьи.

Единственный метод борьбы с внедрением вредоносного кода при вводе данных — подвергать любые данные, получаемые программой от источника, не вызывающего доверия, *тщательной проверке* перед их использованием. Этот подход аналогичен принципу защиты периметра сети, рассмотренному в главе 27: вся информация, поступающая извне доверенного периметра, должна тщательно фильтроваться. Фильтрацию вводимых данных могут выполнять программа или файервол, работающий на прикладном уровне, а также система обнаружения вторжений (ISD) хоста. В идеале фильтрацию выполняют все три компонента: программа лучше всего знает специфику вводимых данных и возможные угрозы, в то время как файервол и ISD могут выполнять более общие проверки для определенного типа угроз.

Троянские программы

Троянские программы, или трояны (trojan), — это разновидность вредоносных программ, которые наносят ущерб системе, маскируясь под какие-либо полезные приложения.

Троянские программы могут быть отнесены к самому простому по реализации виду вредоносных программ, применяя в качестве прикрытия знакомые пользователю приложения, с которыми он работал и раньше, до появления в компьютере «троянского коня», либо принимая вид нового приложения, которое пытается заинтересовать пользователя-жертву какими-то своими якобы полезными функциями. Однако суть троянской программы в обоих случаях остается вредительской: она может уничтожать или искажать информацию на диске, передавать данные (например, пароли) с «зараженного» компьютера на удаленный компьютер хакера, приводить в неработоспособное состояние установленное на атакованном компьютере программное обеспечение, участвовать в проведении DoS-атак на другие удаленные компьютеры. Так, одна из известных троянских программ AIDS TROJAN DISK7, разосланная нескольким тысячам исследовательских организаций на дискете, при запуске перемешивала символы в именах всех файлов и заполняла все свободное пространство жесткого диска.

Сетевые черви

Сетевые черви (worm) — это программы, способные к самостоятельному распространению своих копий среди узлов в пределах локальной сети, а также по глобальным связям, перемещаясь от одного компьютера к другому без всякого участия в этом процессе пользователей сети.

Поскольку большинство сетевых червей передаются в виде файлов, основным механизмом их распространения являются сетевые службы, основанные на файловом обмене. Так, червь может рассылать свои копии по сети в виде вложений в сообщения электронной почты или путем размещения ссылок на зараженный файл на каком-либо веб-сайте. Однако существуют и другие разновидности червей, которые для своей экспансии используют более сложные приемы, например, связанные с ошибками («дырами») в программном обеспечении.

Главная цель и результат деятельности червя состоит в том, чтобы передать свою копию на максимально возможное число компьютеров, — новых потенциальных жертв, для поиска которых черви задействуют встроенные в них средства.

Типичная программа-червь не удаляет и не искажает пользовательские и системные файлы, не перехватывает электронную почту пользователей, не портит содержимое баз данных, а наносит вред атакованным компьютерам *потреблением их ресурсов*, например, для рассылки спама или проведения массированной атаки в составе ботнета.

При создании типичного сетевого червя хакер, прежде всего, определяет перечень сетевых уязвимостей, которые он собирается использовать для проведения атак средствами создаваемого червя. Такими уязвимостями могут быть как известные, но не исправленные на некоторых компьютерах ошибки в программном обеспечении, так и пока не известные никому ошибки, которые обнаружил сам хакер. Чем шире перечень уязвимостей и чем более они распространены, тем больше узлов может быть поражено червем. Червь состоит из двух основных функциональных компонентов:

- *Атакующий блок*, состоящий из нескольких модулей (векторов атаки), каждый из которых рассчитан на поражение конкретного типа уязвимости. Этот блок открывает «входную дверь» атакуемого хоста и передает через нее свою копию.
- *Блок поиска целей* (локатор), собирающий информацию об узлах сети, а затем на основании этой информации определяющий, какие из исследованных узлов обладают теми уязвимостями, для которых хакер имеет средства атаки.

Эти два функциональных блока являются обязательными и присутствуют в реализации любой программы-червя. Некоторые черви нагружены их создателями и другими вспомогательными функциями, о которых мы скажем позже.

Упрощенно жизненный цикл червя может быть описан рекурсивной процедурой, состоящей из циклического запуска локатора и атакующего блока на каждом из последующих заражаемых компьютеров (рис. 32.2).

В начале каждого нового цикла червь, базирующийся на захваченном в результате предыдущей атаки компьютере, запускает локатор для поиска и формирования списка узлов-целей, пригодных для проведения каждой из специфических атак, а затем, используя средства атакующего блока, пытается эксплуатировать уязвимости узлов из этого списка. В результате успешной атаки червь копирует все свои программы на «новую территорию» и активирует локатор. После этого начинается новый цикл. На рисунке показано, как червь лавинообразно распространяется по сети. Заражение тысяч компьютеров может занять всего несколько минут. Некоторые виды червей не нападают на уже зараженные и/или подвергающиеся атаке в данный момент узлы. Если же такая проверка не предусмотрена в алгоритме работы червя, то в сети случайным образом могут возникать очаги стихийных DoS-атак. Локатор идентифицирует цели по адресам электронной почты, IP-адресам, характеристикам установленных на хостах ОС, номерам портов, типам и версиям приложений.

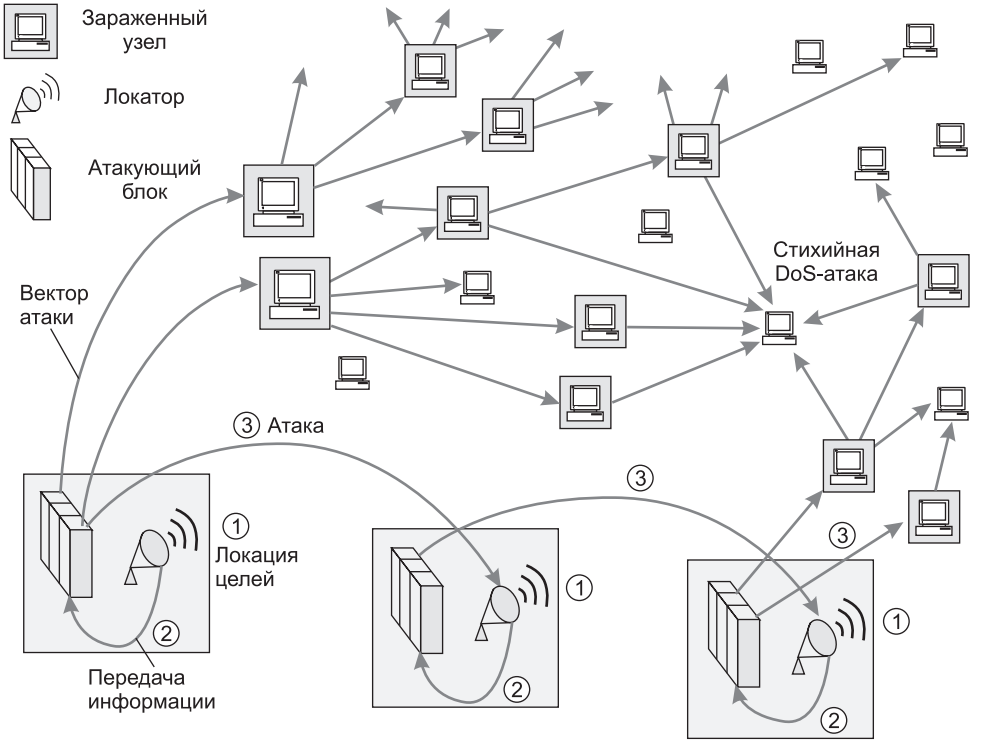


Рис. 32.2. Экспансия червя в сети

Для сбора информации локатор может предпринимать действия, связанные как с поисками интересующих данных на захваченном им в данный момент хосте, так и зондированием сетевого окружения. Простейший способ получить данные локально — прочитать файл, содержащий адресную книгу клиента электронной почты[1]. Помимо почтовых адресов локатор может найти на узле базирования другие источники информации: таблицы конфигурационных параметров сетевых интерфейсов, ARP-таблицы и таблицы маршрутизации. Зная IP-адреса хоста базирования и шлюзов, локатор достаточно просто может определить IP-адреса других узлов этой сети. Для идентификации узлов локатор может также использовать ICMP-сообщения или запросы ping, указывая в качестве адресов назначения все возможные IP-адреса. Для определения того, какие приложения работают на том или ином хосте, локатор сканирует различные хорошо известные номера TCP- и UDP-портов. Определив тип приложения, локатор пытается получить более детальные характеристики этого приложения. Например, пусть некая программа-червь имеет в своем арсенале средства для атаки на некоторые версии веб-сервера Apache. Для поиска потенциальных жертв локатор этого червя зондирует узлы сети, посылая умышленно ошибочные запросы к веб-серверу:

```
GET / HTTP/1.1\r\n\r\n
```

[1] Для коллекционирования почтовых адресов локатор может прибегать и к более интеллектуальным методам, использующим в работе спамеры (о спаме см. далее).

Узел, на котором установлен сервер Apache, отвечает на запрос, как и рассчитывал разработчик червя, то есть сообщением об ошибке, например, такого вида:

```
HTTP/1.1 400 Bad Request
Date: Mon, 23 Feb 2004 23:43:42 GMT
Server: Apache/1.3.19 (UNIX) (Red-Hat/Linux) mod_ssl/2.8.1
OpenSSL/0.9.6 DAV/1.0.2 PHP/4.0.4pl1 mod_perl/1.24_01
Connection: close
Transfer-Encoding: chunked
Content-Type: text/html; charset=iso-8859-1
```

Из этого ответа локатор узнает о том, что на узле установлен веб-сервер Apache версии 1.3.19. Для червя этой информации может быть достаточно, чтобы внести данный узел в число целей.

Собрав данные об узлах сети, локатор анализирует их подобно тому, как это делает хакер при сетевой разведке. Для атаки выбираются узлы, удовлетворяющие некоторым условиям, которые говорят о том, что данный узел, возможно, обладает уязвимостями нужного типа (для них в атакующем блоке есть средства нападения). Понятно, что при таком «предположительном» способе отбора целей не всякая предпринятая атака обязательно приводит к успеху. Неудача рассматривается атакующим блоком червя как штатная ситуация, он просто сворачивает все свои действия, направленные на не поддавшийся атаке узел, и переходит к атаке на следующую цель из списка, подготовленного локатором. Для передачи своей копии на удаленный узел атакующий блок червя часто использует рассмотренную ранее уязвимость *переполнения буфера*. Помимо локатора и атакующего блока (см. выше) червь может включать некоторые дополнительные функциональные компоненты:

- *Блок удаленного управления и коммуникаций* служит для передачи сетевым червям команд от их создателя, а также для взаимодействия червей между собой. Такая возможность позволяет хакеру координировать работу червей для организации распределенных атак отказа в обслуживании. Сетевые черви могут быть использованы и для организации параллельных вычислений при решении таких требующих большого объема вычислений задач, как, например, подбор секретного ключа шифрования или пароля.
- *Блок управления жизненным циклом* может ограничивать работу червя определенным периодом времени.
- *Блок фиксации событий* используется автором червя для оценки эффективности атаки, реализации различных стратегий заражения сети или оповещения других пользователей о повреждениях, нанесенных их компьютерам. Результатом работы данного блока может быть, например, список IP-адресов успешно атакованных машин, посланный хакеру в виде файла или сообщения электронной почты.

Вирусы

Вирус (virus) — это вредоносный программный фрагмент, который может внедряться в другие файлы .

Стремление злоумышленника сделать код вируса как можно более коротким часто ограничивает логику работы вируса очень простыми решениями, которые, однако, иногда приводят к весьма разрушительным последствиям. Так, например, один из реально су-

ществовавших вирусов, состоящий всего из 15 (!) байтов, записывал свою копию поверх других файлов в начало каждого сектора диска, в результате чего система быстро терпела крах. Некоторым утешением в подобных случаях является то, что одновременно с падением ОС прекращает свое существование и вирус.

Вирус может внедрять свои фрагменты в разные типы файлов, в том числе в файлы исполняемых программ (рис. 32.3). При этом возможны самые разные варианты: замещение кода, когда размер инфицированного файла не меняется, вставка вирусного кода целиком в начало или конец исходной программы, замена фрагментов программного кода фрагментами вируса с перестановкой замещенных фрагментов и без перестановки и т. д. и т. п. Более того, код вируса может быть зашифрован, чтобы затруднить его обнаружение антивирусными программами.

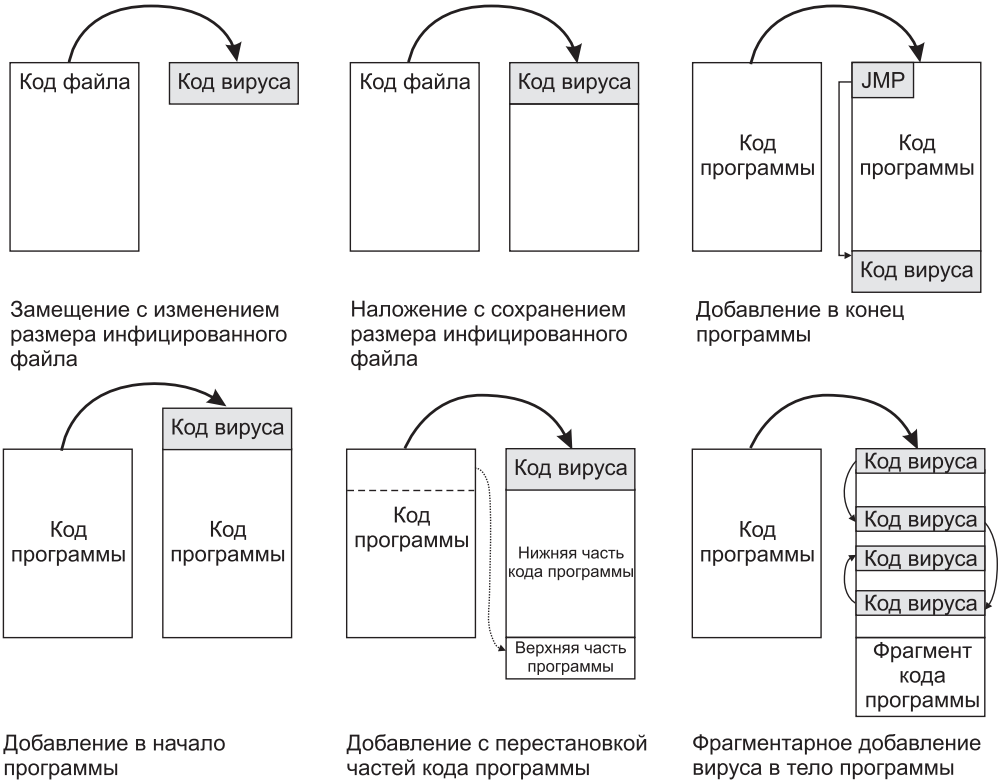


Рис. 32.3. Различные варианты расположения кода вируса в зараженных файлах

В отличие от червей, вирусы (так же, как и троянские программы) не содержат в себе встроенного механизма активного распространения по сети и способны размножаться *своими силами* только в пределах одного компьютера.

Как правило, передача копии вируса на другой компьютер происходит с участием пользователя. Например, пользователь может записать свой файл, зараженный вирусом, на сетевой файловый сервер, откуда тот может быть скопирован всеми пользователями, имеющими доступ к данному серверу. Пользователь может также передать другому пользователю

съемный носитель с зараженным файлом или послать такой файл по электронной почте. То есть именно пользователь является главным звеном в цепочке распространения вируса за пределы своего компьютера. Тяжесть последствий вирусного заражения зависит от того, какие вредоносные действия были запрограммированы в вирусе злоумышленником. Это могут быть мелкие, но раздражающие неудобства (замедление работы компьютера, уменьшение размеров доступной памяти, трата рабочего времени на переустановку приложений) или серьезные нарушения безопасности: утечка конфиденциальных данных, разрушение системного программного обеспечения, частичная или полная потеря работоспособности компьютерной сети.

Программные закладки

Программная закладка — это встроенный в программное обеспечение объект, который при определенных условиях (входных данных) инициирует выполнение не описанных в документации функций, позволяющих осуществлять несанкционированные воздействия на информацию. Функции, описание которых отсутствует в документации, называют **недекларированными возможностями**.

Программные закладки могут выполнять различную вредоносную работу, в частности:

- шпионить за действиями пользователя и передавать эту информацию на определенный сервер — это так называемые **шпионские программы** (spyware);
- получать доступ к конфиденциальной информации;
- искажать и разрушать данные.

В то же время недекларированные возможности программы не обязательно являются вредоносными. Так, они могут быть дополнительными функциями, включенными в программу для отладки, но не имеющими описания для рядовых пользователей. Это могут быть и забытые функции, особенно если речь идет о программной системе, в разработке которой участвовали десятки программистов. Существует также класс недекларированных возможностей программы, внедряемых в нее для развлечения пользователя (и самих программистов тоже), — так называемые «пасхальные яйца» (Easter Eggs). «Пасхальное яйцо» прерывает нормальную работу пользователя, который, возможно, устал рассматривать ячейки таблицы своего документа, и радостно приветствует его интересной картинкой, сообщением, а то и приглашением поиграть в игру. Авторы наблюдали однажды такое «пасхальное яйцо» в заставке экрана «Трубы» (3D Pipes) ОС Microsoft Windows — на экране на несколько минут среди труб появился очень симпатичный чайник. Появился, исчез, и больше никогда мы его не видели, хотя «Трубы» долго работали на наших компьютерах.

Антивирусные программы

Антивирусные программы давно стали необходимым атрибутом жизни любого пользователя. На домашних компьютерах работают индивидуальные пакеты антивирусной защиты, на предприятиях — корпоративные пакеты, состоящие из клиентской программы и сервера, рассылающего обновления. Антивирусные программы используют различные методы для обнаружения вредоносных кодов в файлах, сообщениях электронной почты или HTML-страницах.

Вирус (будем так обобщенно называть далее любой вредоносный код) определенного типа имеет характерную последовательность программных кодов, которая его с какой-то степенью вероятности идентифицирует. Эта последовательность кодов называется **сигнатурой** (подписью) вируса. Чтобы обнаружить вирус, антивирусная программа должна иметь библиотеку сигнатур. Постоянное обновление этой библиотеки является одной из самых главных проблем любой компании, выпускающей антивирусное программное обеспечение. Сервер корпоративной антивирусной системы периодически рассылает обновленные версии такой библиотеки своим клиентам.

Метод сигнатур является основным методом обнаружения вирусов, но обладает принципиальным недостатком — *неспособностью обнаружить новый тип вируса*. Кроме того, разработчики вирусов прибегают к *маскировке* сигнатур, что приводит к нераспознаванию вируса. Для этого, например, злоумышленник может использовать *полиморфический код*, когда код изменяет сам себя во время выполнения, это, естественно, приводит к тому, что у него нет постоянной сигнатуры.

Антивирусные программы используют также **эвристические методы**, которые пытаются выявить вирус на основе структуры его кода или его поведения, не имея точной сигнатуры кода, но используя некоторые обобщенные признаки подозрительной структуры кода (*статический анализ*) или подозрительного поведения (*динамический анализ*).

Для безопасного анализа поведения анализируемой программы она помещается в изолированную виртуальную среду, например, в среду отдельной виртуальной машины или же созданной программной песочницы[1], ограждающей систему от опасных действий программы. В этом случае действия вируса не могут причинить вред основной операционной среде компьютера. Помещение анализируемой программы в специальную защищенную среду является затратным как по ресурсам, так и по времени. Существует более эффективный, хотя и более рискованный подход, когда анализируемой программе разрешают пробное выполнение в рабочей среде, при этом антивирусное программное обеспечение следит за всеми ее действиями и при необходимости блокирует их, не давая нанести ущерб рабочей среде. При обнаружении вируса антивирусная программа помещает зараженную программу в карантин и уведомляет об этом пользователя, который принимает решение об удалении зараженной программы или же, если это возможно, удалении из нее вируса.

ПРИМЕЧАНИЕ

Антивирусные программы работают в пространстве ядра, поэтому сами могут причинить ущерб операционной системе из-за своих ошибок. Зафиксированы случаи, когда под видом антивирусной программы пользователям предлагалось вредоносное программное обеспечение.

Ботнет

Бот — это программа, которая выполняет некоторые автоматические (часто интеллектуальные) действия по командам удаленного центра управления.

Бот является *программным роботом*, способным реагировать на возникающую ситуацию и полученные извне команды некоторыми действиями — протоколированием сообщений (полезный бот ведет архив чатов), отправкой сообщений, например, поддержанием «раз-

[1] Песочница (sandbox) представляет собой механизм жесткого контроля набора ресурсов (оперативной памяти, места на диске и др.) и системных сервисов, доступных подозрительной программе.

говора» с удаленным собеседником или же участием в DDoS-атаке на какой-то сайт или сеть. Бот может, например, распознавать определенный, заданный ему хозяином контекст в дискуссии пользователей социальных сетей интернета (Livejournal, Facebook) и стать ее участником, выдавая те или иные сообщения. Бот обычно находится в следящем режиме, анализируя сообщения и ожидая команды из центра управления или возникновения заранее определенной ситуации.

Боты проникают в удаленные компьютеры нелегально как вирусы, черви или «троянские кони». Пользователь может не знать, что его компьютер заражен ботом, потому что компьютеру этого пользователя бот не причиняет вреда — его цели находятся где-то в интернете. Обычно злоумышленник заражает кодом бота несколько компьютеров, используя различные известные уязвимости ОС и приложений, а затем уже код бота, подобно сетевому червю, пытается заразить как можно больше машин.

Зараженную ботом машину иногда называют **зомби**. Группа согласованно работающих ботов называется **ботнетом** (botnet) или **сетью ботов**. Боты часто управляются централизованно, из одного или нескольких центров, являющихся *серверами сети ботов*. Возможны и более сложные зависимости между ботами одной сети с иерархическими или одноранговыми схемами взаимодействия. Для управления ботами центр управления использует различные протоколы, одним из наиболее распространенных является протокол **IRC** (Internet Relay Chat), позволяющий передавать мгновенные сообщения (чат).

Так как хозяин ботнета точно не знает, какие именно машины оказались зараженными кодом бота, для распознавания компьютеров-зомби используются методы сетевого сканирования, например сканирование портов, если код бота слушает определенный порт TCP. Жертвы зомбирования могут составить внушительную армию, способную претворить в жизнь мощную DDoS-атаку, распространить огромное количество спама или осуществить массовый сбор персональных данных. Заметим, что ботнет может работать и как наемная армия — ее «командир» может предоставлять услуги своей сети третьим лицам.

Одним из инцидентов, связанных с пресечением вредоносной деятельности ботнета, была операция, проведенная компанией Microsoft совместно с ФБР в июне 2013 года по разрушению центров управления ботнетами, зараженными *вирусом Citadel*. Этот вирус фиксирует нажатия клавиш на компьютере и передает информацию в свой центр управления. В результате проведенной операции было выявлено и разрушено 1462 центра управления, каждый из которых контролировал свой ботнет. Сообщалось, что эти сети причинили ущерб почти 5 миллионам пользователей на сумму свыше полмиллиарда долларов.

Безопасность веб-сервиса

Веб-браузер с его графическим интерфейсом является основным средством доступа пользователя к большинству сервисов интернета: сайтам новостей, разнообразным справочникам, библиотекам, интернет-магазинам, онлайновым банкам, социальным сетям, таким как Facebook, Twitter, LiveJournal, ВКонтакте, облачным хранилищам информации и приложениям. Даже такие консервативные устройства, как сетевые маршрутизаторы и коммутаторы, стали поддерживать административный доступ посредством веб-интерфейса[1]. Поэтому

[1] Это в основном относится к домашним маршрутизаторам, рассчитанным на администратора-неспециалиста, которому веб-интерфейс представляется гораздо более удобным, чем командная строка.

справедливым будет сказать, что основная часть информации поступает в клиентский компьютер через веб-браузер, а, как отмечено, именно вводимые данные представляют собой главную угрозу для программного обеспечения компьютера. Через веб-браузер попадает в ваш компьютер большинство вредоносных кодов (вирусы, черви и троянские программы), а также назойливые программки, размещающие рекламные объявления на просматриваемой странице без вашего согласия.

Безопасность веб-браузера

Особенностью защиты веб-службы является то, что такая защита требует решения двух достаточно независимых задач:

- обеспечение безопасности программных и аппаратных ресурсов компьютера, на котором эта служба выполняется;
- обеспечение приватности того лица, которое этой службой пользуется.

Приватность некоторого лица — это требование неприкосновенности частной жизни этого лица, включающая запрет на сбор, хранение, использование и распространение информации о его частной жизни без его согласия.

За время, прошедшее с момента появления первого браузера, разработчики браузеров накопили большой опыт в борьбе со злоумышленниками, и меню *приватности* и *безопасности* современного браузера включает много опций. Рассмотрим, что стоит за этими опциями, какие риски они стараются снизить и за счет каких средств.

Приватность и куки

Популярность интернета негативно повлияла на приватность его пользователей. Потенциально все действия пользователя в интернете — посещенные сайты, просмотренные страницы, запросы поиска — могут быть зафиксированы и проанализированы, и антитеррористические службы, а также службы маркетинга торговых предприятий активно этим занимаются.

Веб-серверы ведут журналы посещений своих сайтов с запоминанием IP-адресов клиентов и предоставляют эти данные владельцам сайтов в удобной форме. Однако анонимность в этих журналах до какой-то степени сохраняется, особенно если адрес назначен провайдером динамически.

Браузеры также ведут журналы посещения сайтов и страниц. И если на веб-сервере данные о ваших посещениях, скорее всего, растворились бы в общей статистике, то на вашем компьютере (если он не используется вместе с другими сотрудниками или посетителями кафе или гостиницы) сохраняется история именно ваших посещений и интересов (последнее — в виде запросов к поисковым машинам). Поэтому теперь конфискация компьютера и просмотр журнала истории браузера — одно из первых действий следователя при расследовании дел в отношении подозреваемой личности. В то же время все современные браузеры позволяют пользователю достаточно детально управлять журналом истории посещений, который хранит как адреса посещенных сайтов и страниц, так и кэшированные страницы этих сайтов.

Угрозу приватности несут также куки. **Куки** (cookies — печенье) представляет собой небольшой фрагмент текстовых данных, которым обмениваются веб-сервер и браузер. Куки,

относящийся к некоторому сеансу браузера с сервером, содержит информацию о текущем состоянии этого сеанса, аутентификационные данные и персональные настройки клиента, а также уникальный для сервера номер сеанса. В течение всего сеанса куки сохраняется *на стороне браузера*. При установлении соединения сервер генерирует содержимое куки и передает его браузеру. Веб-браузер, получив текст куки от веб-сервера, сохраняет его в виде файла. В течение всего сеанса пользователя, а возможно, и при всех повторных обращениях данного пользователя к данному сайту, браузер передает куки серверу в том же виде, в каком он его получил в последнем ответе сервера. Тем самым достигается эффект запоминания состояния сеанса, причем состояние запоминается на стороне клиента.

Веб-сервер обычно применяет данные куки пользователя для его же (пользователя) удобства, например, интернет-магазины обычно хранят в куки карту покупок пользователя, в них также может храниться история навигации пользователя по страницам сайта. Типичной информацией, помещаемой веб-сервером в куки, является идентификатор сеанса пользователя (SID), на основе которого связываются воедино отдельные запросы пользователя. Даже в случае работы по протоколу HTTP 1.1, который поддерживает длительные TCP-сеансы, эти сеансы могут прерываться из-за временной неактивности пользователя, так что объединение отдельных фрагментов сеанса (с тем, чтобы он представлялся пользователю единым) полезно для индивидуального обслуживания пользователя.

Куки бывают *постоянными* (они хранятся в файловой системе ОС и имеют длительные сроки действия) и *временными* (их браузер хранит в оперативной памяти и удаляет после своего закрытия).

Куки имеют не только срок, но и *область действия* — она задается доменным именем сайта, который создал куки. Браузер не передает куки сайту с другим доменным именем, но так как доменное имя может быть задано не для конкретного сайта, а для некоторого домена, то есть, например, не для www.cisco.com, а для cisco.com, то куки могут иметь более широкую область действия, чем один сайт.

Так как куки представляют собой текстовые файлы, то угрозы безопасности для пользователя они не представляют (за исключением случая, когда в них содержится аутентификационная информация пользователя — этот случай рассматривается в следующем разделе). Вирусы и другие вредоносные коды с помощью куки не распространяются, так что бытующее мнение, что куки могут заразить компьютер клиента, не соответствует действительности. В то же время куки могут повредить вашей *приватности*, особенно если в них помещается чувствительная личная информация — данные ваших карт покупок, формы запросов с вашими именем и фамилией, адресом и т. д. Некоторые сайты используют куки третьих сторон, например рекламных компаний. Таким образом, с помощью куки ваши предпочтения становятся известны большому количеству сайтов. Самым простым способом защиты своей приватности является полный запрет на прием куки от любых сайтов, но при этом вы можете лишиться некоторых удобств, основанных на использовании куки, например тех или иных дополнительных услуг интернет-магазина. Поэтому браузеры оставляют пользователю возможность решать, от каких сайтов он запрещает принимать куки, а от каких, наоборот, разрешает.

Протокол HTTPS

Веб-браузер для взаимодействия с веб-сервером по умолчанию использует протокол HTTP без дополнительных мер по обеспечению основных свойств безопасных коммуникаций, то есть аутентификации сторон, а также конфиденциальности, доступности и целостности

данных. Естественно, это создает значительные риски безопасности при работе с сайтами интернета.

Так, при перехвате злоумышленником незащищенных HTTP-пакетов, циркулирующих между веб-браузером и веб-сервером, вполне возможны атаки вида *«человек посередине»*. Одной из разновидностей этой атаки является *захват сеанса*, при котором пользователь аутентифицируется на веб-сервере с помощью своего имени и пароля, а затем веб-сервер рассматривает куки, передаваемые в сообщениях браузера, как свидетельство того, что очередной запрос пришел от аутентифицированного пользователя, и продолжает сеанс без повторного запроса пароля. Понятно, что такой способ аутентификации пользователя в случае множественных сеансов протокола HTTP 1.0 или разрыва по какой-то причине длительного сеанса протокола HTTP 1.1 предоставляет злоумышленнику хорошую возможность для захвата сеанса. Для этого ему достаточно перехватить HTTP-запрос, содержащий куки, и затем посылать свои запросы от имени легального пользователя на соответствующий веб-сервер.

Другим вариантом атаки «человек посередине» является атака *повторения*, когда злоумышленник повторяет перехваченные запросы легального пользователя, возможно, несколько модифицируя их. Например, перехватив запросы сеанса пользователя с его банком, злоумышленник может инициировать повторный перевод денег, но теперь уже на свой счет.

В упомянутых примерах злоумышленник использовал уязвимости процесса *аутентификации* пользователя. Очевидно, что прослушивание открытого трафика между браузером и веб-сервером может также нарушить *конфиденциальность* данных и их *целостность*, если злоумышленник по какой-то причине внесет какие-то изменения в данные. Злоумышленник может также нарушить *доступность* данных, просто отбрасывая ответы веб-сайта.

Основным способом обеспечения перечисленных свойств безопасности данных, циркулирующих между веб-браузером и веб-сайтом, является использование **безопасного протокола передачи гипертекста** (Hypertext Transfer Protocol Secure, **HTTPS**) вместо HTTP. Словосочетание «протокол HTTPS» не вполне корректно, поскольку аббревиатура HTTPS подразумевает *совместно работающую пару протоколов: HTTP и TSL*. Тем не менее название HTTPS прижилось, и пользователь должен его употреблять, когда собирается инициировать защищенное соединение с веб-сервером, например, вводя адрес https://www.cisco.com. В HTTPS-соединении по умолчанию применяется порт 443 вместо порта 80 в HTTP-соединении.

В HTTPS-соединении сам протокол HTTP, работающий поверх протокола TSL, остается неизменным. Все атрибуты безопасности коммуникаций — аутентификация, конфиденциальность и целостность — обеспечиваются протоколом защищенного канала TSL.

Остановимся на некоторых особенностях аутентификации при работе веб-службы. Как вы помните, аутентификация в протоколе TSL основана на цифровых сертификатах. Поэтому при обращении веб-браузера к веб-серверу по протоколу HTTPS каждая из сторон должна иметь подписанный центром сертификации сертификат, достоверность которого можно проверить по цепочке доверия, ведущей к одному из доверенных корневых центров сертификации.

Производители с каждой копией своего браузера поставляют так называемый *встроенный цифровой сертификат*, который может применяться для аутентификации данного браузера. Этот сертификат не аутентифицирует пользователя, работающего с браузером, а служит только для создания защищенного канала при передаче данных между браузером

и веб-сервером. В то же время пользователь может запросить *личный цифровой сертификат* у некоторого центра сертификации и установить его соответствующим образом в своей ОС, указав, что он должен применяться для логического входа. В таком случае вход в веб-сервер, требующий аутентификации, может происходить не на основе имени и пароля пользователя, а с помощью этого сертификата, который поставляется браузером серверу по запросу последнего.

Аутентификация сервера при установлении HTTPS-соединения всегда выполняется на основе *цифрового сертификата сервера*, получаемого владельцем сервера. Этот сертификат подтверждает, что данный веб-сервер имеет определенные (одно или несколько) доменные имена. Браузер обычно уведомляет пользователя о том, что сертификат сервера по какой-то причине является недействительным, оставляя на усмотрение пользователя окончательное решение — отказаться от соединения или все же установить его. Иногда сложный механизм проверки аутентичности сервера работает вхолостую, поскольку пользователи недооценивают угрозы со стороны «невыясненных» веб-серверов и предпочитают действовать на свой страх и риск.

(S) *Проверка действительности сертификата веб-сервера*

Безопасность средств создания динамических страниц

Современные браузеры поддерживают разнообразные средства создания динамических страниц. Все они представляют собой программные коды, полученные извне, и, следовательно, несут риски, связанные с несанкционированным воздействием на клиентский компьютер, начиная с чтения конфиденциальных данных и удаления файлов пользователя до разрушения ОС. Из соображений безопасности пользователи должны очень серьезно относиться к любому предложению веб-сайта установить новую *надстройку* или *вставку*, чтобы, например, лучше проигрывать видеоклип определенного формата или же быстрее загружать файлы. Очень может быть, что, помимо своей основной функции, такая программа будет заниматься и какой-то побочной деятельностью, наносящей вред вычислительной среде пользователя, например, фиксировать нажатия клавиш клавиатуры и передавать их злоумышленнику.

Менее страшны вставки и надстройки, изменяющие параметры и внешний вид браузера так, чтобы заставить пользователя посещать определенные сайты, обращаться к определенным поисковым системам, ориентированным на рекламу, и пользоваться определенными программами. Этот вид вредоносного программного обеспечения получил название *рекламных вирусов* (Adversary Ware, *AdWare*). Избавиться от паразитов бывает непросто — они глубоко встраиваются в ОС и часто не удаляются обычными средствами браузера. Наибольшую опасность для браузера представляют *ActiveX-объекты*, действия которых не ограничены никакими рамками: они могут читать, создавать и удалять файлы, выполнять любые системные действия. Компания Microsoft и другие производители программного обеспечения снабжают свои ActiveX-объекты цифровой подписью, поэтому браузер должен принимать только ActiveX-объекты, подписанные вызывающим доверие разработчиком.

Разработчики *JavaScript-сценариев* и *Java-апплетов*, также применяющихся для создания динамических страниц, встроили в них средства безопасности, что значительно снижает

риски, связанные с их использованием. Браузеры позволяют пользователям управлять процессом создания динамического содержания страницы. Так, пользователь может запретить выполнять ActiveX-объекты или Java-апплеты либо разрешить их выполнение только для доверенных сайтов, список которых он составляет сам.

Безопасность электронной почты

Аналогично защите веб-службы, защита электронной почты также может осуществляться в двух направлениях — обеспечение приватности пользователя (например, конфиденциальности переписки) и обеспечение безопасности ресурсов компьютера (ОС, приложений).

Угрозы приватности почтового сервиса

Пользователи, обменивающиеся сообщениями электронной почты через интернет, должны принимать во внимание наличие следующих угроз:

- спуфинг имени отправителя — злоумышленник выдает себя за другого пользователя;
- спуфинг почтовых серверов — сервер предъявляет при передаче сообщения ложное имя домена;
- модификация сообщения — искажение или отбрасывание сообщения (то есть нарушение целостности или доступности сервиса);
- утечка информации — чтение сообщения злоумышленником (нарушение конфиденциальности);
- нарушение последовательности сообщений;
- нарушение свойства неотказуемости — отказ отправителя от факта отправки письма, отказ почтового сервера от факта приема письма, отказ получателя от факта получения письма;
- спам — засорение почтовых ящиков пользователей письмами, которые пользователи не просили или же не ожидали получить (обычно спам состоит из рекламных сообщений);
- фишинг — электронное письмо обычно является первым этапом фишинга (напомним, что целью такой атаки является завладение учетными данными пользователя для последующего применения, например для снятия денег со счета, в электронных платежах и т. п.). Такое электронное письмо может выглядеть очень похожим на «настоящее», то есть иметь все атрибуты оформления письма некоторого банка или солидной организации и содержать просьбу обновить свой пароль по приводимой ссылке. Второй этап фишинга выполняет веб-сайт, на который попадает пользователь, перейдя по ссылке;
- нарушение приватности пользователя за счет сбора метаданных почтового сервиса.

Все перечисленные угрозы — следствие того, что изначально почтовая служба интернета, основанная на протоколе SMTP, не поддерживала никаких механизмов защиты почтового обмена — текст сообщения в SMTP-пакетах передавался в открытом виде и его легко было прочитать и модифицировать. Спуфинг отправителя также являлся очень простым делом — почтовый клиент злоумышленника или же его почтовый сервер помещали туда любое имя, требуемое для обмана получателя. Факт такой подмены обнаружить трудно, так как имена пользователей не хранятся в DNS и проверить соответствие IP-адреса имени

этим путем невозможно, а аутентификация отправителя в протоколе SMTP предусмотрена не была (получатель аутентифицировался паролем при получении сообщения). Отсутствие аутентификации отправителя делало сложным обеспечение *неотказуемости* — всегда можно было отказаться от факта отправки письма, сославшись на спуфинг отправителя, мол, это кто-то другой его написал, а указал меня в качестве отправителя. Квитанция о прочтении письма тоже не является в таких условиях достоверной — ее мог сгенерировать злоумышленник, преследуя какую-то свою цель. Отправителю спама также легко было отказаться от авторства рассылки.

К сожалению, применение прошедшего времени в описании такой грустной картины не совсем оправданно — сплошь и рядом электронная почта интернета используется в своем первозданном виде, хотя за долгие годы существования сервиса разработаны различные стандарты безопасности электронной почты. Велика и инерция масштабной распределенной системы интернет-почты — существует огромное количество почтовых серверов, работающих под управлением старых версий программного обеспечения, не поддерживающего новые стандарты, или же под управлением новых версий, в которых новые функции защиты не активированы администраторами. Далее рассмотрены несколько стандартов безопасности почты интернета, направленные на снижение рисков, связанных с ее работой.

Аутентификация отправителя

Существует несколько методов аутентификации отправителя:

- ограничение отправителей провайдером услуг;
- аутентификация отправителя провайдером услуг;
- аутентификация отправителя на основе его личного сертификата.

Ограничение отправителей провайдером услуг не является в строгом смысле аутентификацией. Этот способ основан на том, что почтовый сервер провайдера принимает по протоколу SMTP только те письма, которые отправляются клиентами этого провайдера. Принадлежность отправителя к клиентам провайдера проверяется по его IP-адресу — адрес должен принадлежать пулу адресов, которым провайдер владеет и которые он выделяет своим клиентам. Некоторые провайдеры поступают еще строже — они не разрешают своим клиентам пользоваться чужими почтовыми серверами для отправки писем, блокируя соединения на порт 25 от клиентских компьютеров, если они направлены не к почтовому серверу провайдера. То есть провайдер не только блокирует чужих пользователей, но и не разрешает своим пользователям обращаться к почтовым услугам других провайдеров. Тем самым осуществляется взаимная защита провайдеров от чужих пользователей (а также привязка пользователей к провайдеру, что преследует чисто коммерческие цели). Этот метод не гарантирует получателю аутентичности отправителя, но защищает провайдера от спама, отправляемого чужими пользователями.

Аутентификация отправителя провайдером услуг. Расширение протокола SMTP — **SMTP AUTH** — описывает процедуру аутентификации пользователя при отправке сообщения агентом пользователя серверу провайдера почтовых услуг. В соответствии с этим расширением почтовый сервер и агент пользователя в начале SMTP-сеанса договариваются о методе аутентификации. В список возможных методов, в частности, входят: открытый пароль (обычно передается по защищенному каналу TSL), аутентификация на основе слова-вызова и др.

Аутентификация пользователя первым сервером почтовой системы решает многие проблемы — защищает провайдера от спама, позволяет при необходимости решить проблему неотказуемости отправителя. Но при дальнейшей передаче информация об аутентичности пользователя теряется, поэтому отправитель должен полагаться на добросовестность провайдера, под чьим административным управлением находится почтовый сервер. Даже если провайдер достоин доверия, этот факт не исключает атаки «человек посередине», когда кто-то перехватывает сообщение по пути к почтовому серверу получателя и изменяет имя отправителя.

Аутентификация отправителя на основе его личного сертификата. Этот способ аутентификации работает «из конца в конец», так как сообщение подписывается цифровой подписью отправителя, чей открытый ключ находится в его личном сертификате. Возможность включения цифровой подписи в качестве части сообщения описана в расширении S/MIME и предусматривает использование различных стандартов цифровой подписи, например **PKCS-7** компании RSA или **PGP (Pretty Good Privacy)**. Аутентификация на основе цифровой подписи отправителя решает несколько задач:

- получатель может проверить аутентичность отправителя и целостность сообщения;
- отправитель не может отказаться от факта отправки письма;
- подпись квитанции о получении/чтении письма делает невозможным отказ получателя от факта получения письма.

Цифровая подпись в расширении S/MIME занимает две части сообщения:

- в первой части описывается используемый стандарт цифровой подписи (протокол) и примененная хеш-функция;
- во второй части, которая является приложением, находится сама цифровая подпись, охватывающая все части сообщения вместе с их заголовками.

В варианте PKCS-7 частью цифровой подписи S/MIME является также цифровой сертификат, выданный одним из сертифицирующих центров, входящих в иерархию PKI, и удостоверяющий принадлежность открытого ключа отправителю, указанному в заголовке почтового сообщения. Вот пример сообщения, подписанного по стандарту PKCS-7 почтовым клиентом Microsoft Windows Mail 6.0 и принятого почтовым клиентом Apple Mail 6.6 (сообщение представлено в режиме Raw Source программы Apple Mail 6.6, показывающим все MIME-элементы сообщения):

```
Return-path: <natalia@olifer.co.uk>
Envelope-to: victor@olifer.co.uk
Message-ID: <5ED892093E784C5D9BFD602759D9A7C5@natashaPC>
From: <natalia@olifer.co.uk>
To: "victor" <victor@olifer.co.uk>
Subject: secure email
Date: Sat, 9 Nov 2013 11:05:18 -0000
MIME-Version: 1.0
Content-Type: multipart/signed;
    protocol="application/x-pkcs7-signature";
    micalg=SHA1;
    boundary="----=_NextPart_000_0017_01CEDD3B.940A3930"
X-Priority: 3
X-MSMail-Priority: Normal
X-Mailer: Microsoft Windows Mail 6.0.6002.18197
X-MimeOLE: Produced By Microsoft MimeOLE V6.0.6002.18463
```

```
This is a multi-part message in MIME format.
------=_NextPart_000_0017_01CEDD3B.940A3930
Content-Type: multipart/alternative;
    boundary="----=_NextPart_001_0018_01CEDD3B.940A3930"
------=_NextPart_001_0018_01CEDD3B.940A3930
Content-Type: text/plain;
    charset="iso-8859-1"
Content-Transfer-Encoding: quoted-printable
Hi,=20
It is much better to use a secure email correspondence.=20
Enjoy!

------=_NextPart_000_0017_01CEDD3B.940A3930
Content-Type: application/x-pkcs7-signature;
    name="smime.p7s"
Content-Transfer-Encoding: base64
Content-Disposition: attachment;
    filename="smime.p7s"
MIAGCSqGSIb3DQEHAqCAMIACAQExCzAJBgUrDgMCGgUAMIAGCSqGSIb3DQEHAQAAoIITJjCCBDYw
ggMeoAMCAQICAQEwDQYJKoZIhvcNAQEFBQAwbzELMAkGA1UEBhMCU0UxFDASBgNVBAoTC0FkZFRy
dXN0IEFCMSYwJAYDVQQLEx1BZGRUcnVzdCBFeHRlcm5hbCBUVFAgTmV0d29yazEiMCAGA1UEAxMZ
QWRkVHJ1c3QgRXh0ZXJuYWwgQ0EgUm9vdDAeFw0wMDA1MzAxMDQ4MzhaFw0yMDA1MzAxMDQ4Mzha
```

Здесь мы видим обе части цифровой подписи. Первая часть говорит о том, что это сообщение снабжено цифровой подписью:

```
Content-Type: multipart/signed;
protocol="application/x-pkcs7-signature";
micalg=SHA1;
```

Вторая часть (отделена пустой строкой) представляет собой собственно цифровую подпись, форматированную алгоритмом base64, который заменил 8-битные коды подписи ASCII-символами.

На клиенте отправителя был установлен личный сертификат, полученный от компании Comodo. Почтовые клиенты автоматически проверяют подлинность сертификата, с помощью которого получена цифровая подпись PKCS-7. Для пользователя этот этап незаметен — в случае положительной проверки он видит только обычное сообщение (помеченное, как правило, особым значком и сообщением «подписано таким-то»). А вот в случае отрицательной проверки, когда сертификат отправителя по какой-то причине оказался недействительным, пользователю-получателю выводится на экран предупреждение, и решение о том, принять сообщение или нет, остается за ним.

Второй вариант цифровой подписи в стандарте S/MIME использует технологию PGP. Система **PGP** заслуживает особого внимания — именно она была первой системой цифровой подписи и шифрования почтовых сообщений интернета, использующей технику публичных ключей. Одним из основных отличий подходов, применяемых в PGP и PKCS-7 к получению цифровой подписи, является то, что в PGP принадлежность открытого ключа некоторому отправителю должна быть подтверждена заранее, до получения письма от данного отправителя. Открытые ключи отправителей, с которыми получатель поддерживает защищенную переписку, должны храниться в некотором хранилище, доступном почтовому клиенту получателя. При приходе письма от доверенного отправителя клиентская почтовая программа получателя проверяет подлинность цифровой подписи с помощью открытого ключа отправителя, извлекая его из хранилища. Для поддержки операции проверки при-

надлежности открытого ключа некоторому пользователю в PGP вводится понятие **паутины доверия** (Web of Trust). Эта паутина похожа на публичную структуру PKI, так как использует цифровые сертификаты и подразумевает иерархию подписывающих их сущностей, то есть пользователей, которым вы прямо или косвенно (через иерархию доверительных отношений) доверяете. В принципе, пользователь системы PGP волен сам решать, каким образом проверять принадлежность открытого ключа другому пользователю PGP.

Шифрование содержимого письма

Шифрование содержимого письма может происходить как «из конца в конец», так и на отдельных участках маршрута следования письма, например, между агентом пользователя и почтовым сервером, принимающим письма от пользователей.

- Шифрование содержимого письма *«из конца в конец»* предусмотрено спецификацией S/MIME. Она определяет способ шифрования определенной части составного сообщения, шифруемой вместе со своим заголовком.
- Шифрование *на отдельных участках* чаще всего осуществляется средствами защищенного канала, создаваемого между двумя непосредственно общающимися сторонами передачи сообщения. Этот канал может быть IPSec- или TSL-каналом, в зависимости от предпочтений администраторов сетей, в которых расположены эти стороны. Однако такой способ шифрования не гарантирует конфиденциальности сообщения на всем пути от отправителя до получателя, так как какой-то другой участок пути может не использовать защищенный канал, а протокола общей координации участников распределенной схемы передачи писем пока не существует.

Как и в случае цифровой подписи, для передачи шифрованного сообщения требуются две части — в первой части описывается факт шифрования и его способ, а вторая часть является приложением, в котором находится зашифрованная исходная часть сообщения. Спецификация S/MIME предусматривает использование PKCS-7 и PGP.

Защита метаданных пользователя

Метаданными электронной почты называют некоторые характеристики сообщений, которые, не передавая самого содержимого сообщения, определяют адресатов переписки и некоторые другие обстоятельства этого процесса. Точнее, к метаданным электронной почты относят:

- имя отправителя, его почтовый адрес и его IP-адрес;
- имя получателя, его почтовый адрес и его IP-адрес;
- тип данных и их кодировки;
- уникальный идентификатор сообщения и связанных с ним сообщений;
- дату, время и временную зону отправки и получения сообщения;
- форматы заголовков сообщения;
- тему письма;
- статус сообщения;
- запрос на подтверждение получения и открытия письма.

Как видно из описания, сбор метаданных почтового сервиса может дать детальную картину о деятельности некоторого пользователя, даже если он шифрует свои сообщения. Собрать их достаточно просто. Во-первых, потому что метаданные телекоммуникационных сервисов — почты, мобильной связи, веб-сервиса и др. — законодательствами большинства стран либо совсем не защищаются, либо защищаются в намного меньшей степени, чем собственно данные сообщений сервиса. То есть в то время, как раскрытие содержимого переписки в интернете требует решения суда, *сбор метаданных не считается атакой* и может проводиться беспрепятственно.

Во-вторых, метаданные именно электронной почты легче привязать к определенному пользователю. Метаданные электронной почты хранятся на компьютерах отправителя и получателя (как и сами сообщения), но, что опасно для приватности пользователей, еще и *в журналах почтовых серверов*, которые передавали эти сообщения. Метаданные пользователей почтового сервиса гораздо легче найти на серверах провайдеров, чем метаданные пользователей веб-сервиса, потому что пользователи почты «привязаны» к определенным почтовым серверам, например, они отправляют почту либо через сервер своего домашнего провайдера, либо корпоративный сервер, либо сервер провайдера гостиницы, вокзала или кафе, где они временно находятся, либо через сервер публичной почты, такой как Gmail. Пользователь получает почту также через вполне определенный сервер, на котором у него имеется учетная запись. Эта ситуация не похожа на веб-сервис, где пользователь может посетить любой сервер интернета, так что найти следы его посещений путем проверки серверов практически невозможно, даже если пользователь регистрировался на некоторых из них.

О ценности метаданных

Вынужденная отставка генерала Давида Петреуса из-за раскрытия любовной связи с его биографом Полой Бродвел подтверждает важность почтовых метаданных. Петреуса нельзя считать человеком, неосведомленным в вопросах информационной безопасности, — после многих лет блестящей военной карьеры, на пике которой он возглавлял штаб вооруженных сил США, командовал объединенной группировкой войск в Афганистане, Петреус был назначен директором ЦРУ. Тем не менее главный шпион Америки понадеялся на то, что анонимный почтовый аккаунт в Gmail будет вполне безопасен для переписки с Полой, если они не будут отправлять с него писем, а только оставлять черновики писем в локальной папке сервера Gmail. Можно, конечно, сказать, что во всем была виновата Пола, которая начала отправлять с этого аккаунта угрожающие письма Джил Келли, другу семьи Петреусов. Джил заявила об этих письмах ФБР, и это инициировало расследование. Так как в деле было замешано имя директора ЦРУ, расследование было проведено тщательно, и выйти на след анонимного пользователя почтового аккаунта помогли почтовые метаданные. Хотя в содержании писем не было никаких «зацепок», позволяющих определить личность автора, агенты ФБР смогли его найти, сопоставив данные логических входов анонима с перемещениями лиц из круга знакомых Петреуса. Выяснилось, что IP-адреса анонима принадлежат нескольким гостиницам, в которых останавливалась Пола точно в те дни, когда аноним входил в свой аккаунт. Этого совпадения оказалось достаточно, чтобы основной подозреваемой стала Пола, а дальнейшие доказательства были уже добыты стандартными способами — обысками дома, личного компьютера и допросами.

Ценность метаданных хорошо понимают спецслужбы, недаром одна из программ NSA, о которых рассказал миру Сноуден, называется телефонной и связана с массовым сбором метаданных мобильных пользователей, благо что законы, охраняющие приватность

в США, запрещают прослушивание телефонных разговоров, но не запрещают собирать метаданные мобильных клиентов.

Спам

Спамом называют рассылку писем большому числу адресатов без их согласия или даже намерения вступить в переписку (по названию постоянно навязываемых посетителям кафе консервов из скетча Монти Пайтона). Является ли рассылка спама преступлением или нет, определяется законодательством конкретной страны. В начале 2000-х во многих странах были приняты акты, определяющие, что является спамом и какие наказания применять за его рассылку. Однако принятие этих актов только незначительно снизило процент спама, в общем потоке электронных писем он по-прежнему очень высок и достигает 80–85 %. Это связано с тем, что определение спама является достаточно безобидным. Например, массовая рассылка не считается спамом, если в письме ясно указана его рекламная цель, а получатель имеет возможность отписаться от рассылки. Кроме того, доказать на практике тот факт, что пользователь не давал согласие на получение письма, сложно.

Со спамом борются провайдеры интернета. В «Терминах и условиях» их договоров с пользователями обычно есть пункт, запрещающий пользователю рассылать спам. В интернете существуют так называемые «черные списки» (blacklists) IP-адресов электронной почты и/или доменных имен, с которых рассылался спам и письма с которых рекомендуется блокировать. Наиболее распространенной практикой является ведение черных списков в виде зон системы доменных имен (DNS). Такая практика получила название **DNSBL** (DNS Black Lists). Почтовый сервер провайдера может быть сконфигурирован так, что он автоматически опрашивает какой-либо файл DNS-зоны, содержащий черный список, и блокирует письмо, если адрес или имя его отправителя имеется в списке.

Одним из наиболее часто используемых черных списков спамеров является список некоммерческой компании Spamhouse. В главе 31 рассматривалась мощная DDoS-атака с суммарной интенсивностью трафика в 75 Гбит/с, которой в марте 2013 года был подвергнут веб-сервер этой компании. Атака на Spamhouse была местью одной из компаний, занимающейся рассылкой спама, за включение ее в черные списки. Spamhouse имеет очень мощную распределенную систему DNS-серверов, которые предоставляют по запросу почтовых серверов или клиентов черные списки. Spamhouse ведет несколько таких списков — для спамеров, для хостов, зараженных вирусами, для хостов, не выполняющих аутентификацию при передаче письма на почтовый сервер (это считается нарушением политики безопасности почтового сервиса). Владельцы адресов, попавших в черный список, могут оспорить решение Spamhouse, это нормальная процедура, так как при современном «незащищенном» состоянии почты интернета ошибки при определении источника спама неизбежны.

Атаки почтовых приложений

Текст почтового сообщения на первый взгляд не может причинить вред компьютеру пользователя. Однако гибкость современной почты позволяет злоумышленникам внедрять в сообщения разнообразную информацию, в том числе исполняемые коды, которые уже не являются столь безобидными. Проще всего поместить вредоносный код в *приложение* почтового сообщения. Почтовый клиент при открытии пользователем приложения пере-

дает его одной из программ клиентского компьютера для обработки. Если приложением является исполняемый файл, например файл с расширением .exe, то почтовый клиент передаст его на выполнение ОС.

Расширения выполняемых программ могут быть и другими, например, если это Java-программа или скрипт командного процессора. Исполняемый код может быть также выполнен в виде макроса или скрипта какого-либо документа, например документа MS Word или Excel. Такое приложение вызывает меньше подозрений (ведь это только текст или таблица), но скрипты документов также могут получить доступ к ресурсам компьютера и причинить ему вред. Наконец, вредоносный код может находиться и в *теле сообщения* (если оно написано на языке HTML) в виде JavaScript-скриптов или, что действительно опасно, ActiveX-объектов. Поэтому все почтовые сообщения должны проходить обязательную проверку антивирусной программой на наличие вредоносного кода в приложениях и самом сообщении.

Безопасность облачных сервисов

Облачные вычисления как источник угрозы

Ограниченная подконтрольность провайдера

Модель облачных вычислений существенно отличается от традиционной модели вычислений, используемой сегодня в корпоративных ИС, и это отличие прежде всего сказывается на обеспечении их безопасности. Природа данного отличия довольно проста — вместо того, чтобы строить собственную ИС и управлять ею силами сотрудников предприятия, предприятие начинает пользоваться услугами системы, созданной посторонней организацией-провайдером. При этом организация-провайдер владеет всей инфраструктурой ИС и управляет ею, обеспечивая в том числе безопасность данных, принадлежащих предприятию-клиенту. Сотрудники предприятия-клиента используют свои компьютеры только как терминалы доступа к облаку, а все данные предприятия, включая личные данные сотрудников, хранятся и обрабатываются «где-то там, в облаке». Предприятию-клиенту остается только следовать совету Рональда Рейгана: «доверяй, но проверяй».

Такой революционный переворот в модели вычислений не мог не обеспокоить специалистов по безопасности. Многие из них согласны в том, что облачные вычисления — вещь хорошая, но *отсутствие гарантий безопасности* облачных вычислений сводит на нет все преимущества облака.

Для этих опасений, безусловно, есть основания, однако многое зависит от типа организации-клиента и модели облачных вычислений, которую клиент собирается использовать. Понимание моделей облачных вычислений — необходимое условие для правильной оценки рисков предприятия при переходе на новый тип ИС.

Предприятие-клиент облачных сервисов имеет весьма *ограниченный контроль* над механизмами безопасности своих данных, обрабатываемых виртуальными машинами провайдера и хранящимися в виртуальных хранилищах. Особенно это справедливо для услуг модели SaaS, когда защита всех элементов ИС, включая прикладные программы пользователя, осуществляется провайдером. Предприятие-клиент в этом случае участвует лишь в обеспечении безопасности компьютеров своих сотрудников, которые применяются как

терминалы облачной среды. При этом клиент IaaS-сервиса должен самостоятельно заботиться о безопасности своих приложений — следить за тем, чтобы обновления приложений периодически получались и устанавливались, устанавливать и обслуживать антивирусные программы и программы блокировки спама, выполнять все остальные действия в соответствии с политикой безопасности предприятия, которые относятся к приложениям. Обычно в этой модели конфигурирование средств безопасности ОС также является делом клиента.

В модели PaaS контроль над средствами безопасности верхних уровней разделяется между провайдером и клиентом. В таких условиях клиент должен стараться получить как можно более *полный доступ к средствам аудита провайдера* — к сообщениям и журналам средств безопасности, его виртуального файервола, системы IDS, антивирусных и антиспамовых программ и т. п. Кроме того, полезно также проводить *аудит работы самого провайдера*, привлекая для этого сторонние фирмы, пользующиеся устойчивой репутацией в этой области.

Получение информации от средств защиты провайдера в реальном времени (мониторинг) и доступ к историческим данным этих средств должен быть предусмотрен в *договоре о предоставлении услуг провайдером*. Этот важный документ должен оговаривать все детали взаимоотношений провайдера и клиента, а так как облачные услуги являются новым видом телекоммуникационных услуг, то внимание ко всем его деталям должно быть самое пристальное.

(S) *Соглашение об уровне обслуживания с провайдером облачных сервисов*

Наличие сертификатов на соответствие средств безопасности провайдера популярным программам сертификации также может частично компенсировать отсутствие полного контроля над этими средствами.

Разделение сложной инфраструктуры провайдера

При использовании услуг облачного провайдера вы *разделяете его инфраструктуру* с другими арендаторами, которых не знаете. Разделение ресурсов провайдера осуществляется не на физическом уровне, а с помощью механизмов виртуализации. Значит, злоумышленник может заключить договор с вашим провайдером и попытаться использовать бреши в механизмах виртуализации для получения несанкционированного доступа к вашим данным. Кроме того, нельзя исключать ошибок персонала провайдера, в результате которых виртуальные барьеры могут быть нарушены.

Инфраструктура облачного провайдера намного *сложнее* инфраструктуры стандартного центра данных, что представляет собой дополнительную угрозу безопасности облачных сервисов. Кроме таких стандартных элементов виртуализации, как гипервизоры, виртуальные машины, виртуальные маршрутизаторы и файерволы, подобная инфраструктура включает многочисленные дополнительные компоненты управления: средства самостоятельного динамического выделения ресурсов клиентами, измерители потребления ресурсов, средства управления квотами, нагрузкой, мониторинга качества, услуг и т. п. Кроме того, облачные сервисы могут быть реализованы в облачной среде другого провайдера, что еще более усложняет картину. Обычно администраторы корпоративных ОС и приложений получают доступ к ним через локальную сеть. При использовании облачных сервисов административный *доступ должен выполняться через интернет*, что несет дополнительные угрозы. Необходимо убедиться, что облачный провайдер поддерживает только хорошо защищенные соединения для предоставления административного доступа своим клиентам.

Угроза конфиденциальности персональным данным

При использовании услуг корпоративной сети персональные данные сотрудников хранятся в справочной службе предприятия (например, работающей на основе Microsoft Active Directory) и предприятие несет ответственность за их конфиденциальность в соответствии с законами и правовыми актами. В том случае, когда сотрудники пользуются услугами облачного провайдера, их персональные данные (имена и пароли) хранятся в справочной службе провайдера. Так как ответственность за их конфиденциальность в конечном счете все равно несет предприятие, необходимо убедиться, что провайдер надлежащим образом обеспечивает их конфиденциальность — как при передаче личных данных через интернет, так и при хранении их в разделяемой между арендаторами справочной службе провайдера. Для повышения уровня защиты личных данных можно применять раздельные наборы данных для локальной аутентификации пользователей и их аутентификации у облачного провайдера. Но это довольно громоздкое решение, которое для большой организации может оказаться неработоспособным.

Другим решением является применение схемы федеративной аутентификации — личные данные пользователей хранятся в справочной службе предприятия, а служба аутентификации провайдера взаимодействует со службой аутентификации предприятия через защищенное соединение интернета.

Мультинациональность облачных услуг, законодательство и политика

В мире облачных вычислений существуют различные варианты реализации сервисов в отношении того:

- где данные физически размещены;
- где они обрабатываются;
- откуда происходит доступ к этим данным.

Зачастую эти три точки находятся в разных странах, в каждой из которых действует свое законодательство. В разных странах также могут быть зарегистрированы предприятие-клиент и облачный провайдер. Законодательные аспекты таких многонациональных услуг определены плохо — эта работа находится в начальной стадии. В результате появляется много неясностей во взаимоотношениях провайдеров и клиентов многонациональных облачных услуг, а значит, риски использования этих услуг весьма высоки. Снизить риски, связанные с многонациональными облачными сервисами, можно за счет непосредственного указания в договоре конкретной судебной инстанции определенной страны, которая будет разбирать любые споры между провайдером и клиентом, если они возникнут.

Чтобы облачные вычисления могли успешно развиваться как глобальная, не знающая границ услуга, они должны быть отделены от политики. К сожалению, сегодня законы, принимаемые разными правительствами, часто оказывают негативное влияние на развитие глобального облака. Так, одним из результатов принятия США Патриотического Акта 2004 стало то, что Канада решила не использовать серверы интернета, расположенные на территории США, опасаясь за конфиденциальность данных, которые Канада хранит на своих компьютерах. Разоблачения Сноудена в отношении программы PRISM привели Бразилию к решению построить собственный сегмент интернета с основными сервисами, поддерживаемыми серверами, находящимися на территории Бразилии.

Облачные сервисы как средство повышения сетевой безопасности

Несмотря на то что облачные сервисы принято рассматривать как источник новых угроз безопасности, они могут существенно *улучшить информационную безопасность* предприятия — особенно если это небольшое предприятие, у которого нет специального подразделения, занимающегося безопасностью. Существует несколько преимуществ облачной модели над традиционными подходами к организации вычислений. Некоторые из них следует рассматривать в качестве потенциально применимых, поскольку они предполагают соответствующую корректную организацию процессов управления безопасностью провайдером услуг; другие же являются следствием самой парадигмы облачных вычислений.

Избыточность и резервирование ресурсов

Высокая степень масштабирования ресурсов облачных провайдеров обеспечивает также *высокую доступность* этих ресурсов и, как следствие, данных, обрабатываемых этими ресурсами. Избыточность и резервирование ресурсов являются одними из основных принципов построения облачной среды. Обычно облачный провайдер располагает большим количеством центров данных в разных географических точках, а возможно и странах, при этом реплики данных одного и того же клиента хранятся в нескольких таких центрах. Это требуется как для повышения производительности за счет приближения данных к пользователям (если это данные публичные, предназначенные для всех пользователей интернета), так и для обеспечения доступности данных в случае технических отказов или природных катастроф. Кроме того, избыточность ресурсов вызвана необходимостью обеспечивать эластичность сервисов, то есть возможность быстрого увеличения нагрузки по запросу клиента. В корпоративной сети такую высокую доступность ресурсов обеспечить трудно, и она чаще всего будет экономически неоправданна.

Поглощение DDoS-атак

Распределенная избыточная инфраструктура центров данных облачного провайдера позволяет *эффективно бороться с DDoS-атаками*. Отражение мощной DDoS-атаки является, наверное, наиболее сложной задачей для администратора корпоративной сети, так как фильтрация потока пакетов интенсивностью в десятки, а то и сотни гигабайт в секунду, направленного на единственную копию сервера через канал связи, требует наличия очень производительного файервола. Такие файерволы, специально созданные для отражения DDoS-атак, существуют. Однако даже если предприятие может себе позволить приобрести и установить столь дорогостоящее устройство, остается проблема узкого места, которое представляет собой единственная копия атакуемого сервера (скорее всего, это веб-сервер, возможности которого публиковать открытые данные злоумышленник пытается подавить) и единственный канал связи сервера с интернетом с фиксированной пропускной способностью. Поэтому в начальной стадии атаки, когда администратор еще не успел определить признаки, отличающие пакеты атаки от пакетов легальных пользователей, файервол принципиально не может защитить сервер от атаки, пропуская весь трафик к серверу и тем самым блокируя его работу.

В сети провайдера облачных услуг отразить DDoS-атаку на ресурсы клиента принципиально проще. На рис. 32.4 показана достаточно типичная структура сети облачного провай-

дера с четырьмя центрами данных, рассредоточенными по географическим регионам, при этом в каждом центре имеется сервер с копией данных некоторого клиента. Для баланса нагрузки провайдер применяет маршрутизацию с произвольной (anycast) рассылкой, в результате трафик запросов клиентов направляется ближайшему (относительно метрики маршрутизации) серверу.

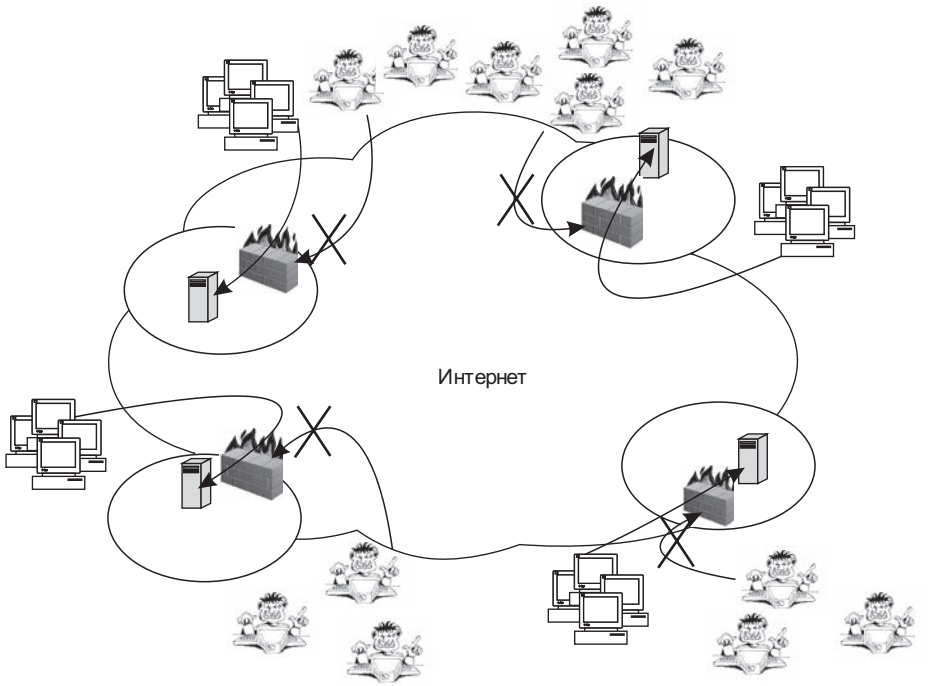


Рис. 32.4. Распределение и поглощение трафика DDoS-атаки инфраструктурой облачного провайдера

При возникновении DDoS-атаки трафик ботов злоумышленника так же распределяется между серверами провайдера, как и трафик клиентов, поэтому атака не может быть такой же эффективной, как в случае единственного сервера и единственной линии доступа к нему. К тому же провайдер, как правило, поддерживает значительный запас пропускной способности линий доступа для успешного обслуживания пиков потребностей клиентов, поэтому «забить» линии доступа облачного провайдера злоумышленнику труднее. Можно сказать, что в начальный период DDoS-атаки инфраструктура облачного провайдера «впитывает» трафик атаки как губка, без значительного ущерба для трафика легальных пользователей. А так как серверы провайдера защищены высокопроизводительными файерволами, то после обнаружения факта атаки и определения признаков, по которым можно отличить трафик атаки от трафика пользователей, администратор провайдера вносит соответствующие изменения в правила файерволов и они начинают блокировать трафик атаки. Внимательный читатель может заметить, что только что прочитанное пояснение вполне подходит к описанию DDoS-атак на корневые DNS-серверы — и это не удивительно, так как механизм поглощения трафика атаки в обоих случаях одинаков.

Квалификация персонала и качество платформы вычислений

Провайдер облачных услуг является крупной организацией (иначе он не сможет обеспечить масштабируемость, эластичность и некоторые другие свойства этой модели) и, как всякая крупная организация, может себе позволить специализацию своих администраторов и операторов во всех областях обеспечения информационной безопасности. В результате специалисты провайдера получают большой опыт в распознавании всего спектра угроз, актуальных в настоящее время, а также в установке и конфигурировании средств отражения этих угроз, таких как файерволы, системы IDS/IPS, антивирусные системы и т. п., что, естественно, повышает безопасность данных клиентов облачных сервисов.

Важно и то, что платформа вычислений (сетевая и компьютерная) может оказаться более качественной у облачного провайдера, чем у предприятия-клиента. Причиной является ее более *высокая степень однородности*, которая определяется тем, что провайдер оказывает одни и те же услуги большому количеству клиентов. Поэтому центры данных провайдера, как правило, строятся на одних и тех же моделях маршрутизаторов, коммутаторов, файерволов, серверов и гипервизоров виртуальных машин. Однородность упрощает управление и сокращает количество ошибок конфигурирования, а значит, делает более простым и эффективным обеспечение безопасности такой платформы.

Другая причина заключается в *масштабности* центров данных и сети провайдера облачных сервисов, что позволяет ему покупать для платформы самые совершенные и производительные компоненты защиты данных, например высокопроизводительный файервол, способный отфильтровывать пакеты интенсивных DoS-атак. Провайдеры облачных услуг стараются сертифицировать свои платформы по различным программам сертификации безопасности, чтобы завоевать доверие клиентов, — это также повышает вероятность того, что используемая платформа обладает высоким качеством.

После выяснения новых видов угроз, связанных с применением публичных облачных сервисов, у корпоративных специалистов по безопасности может возникнуть вопрос: а стоит ли овчинка выделки? Ответ не очевиден, но при его выборе нужно принимать во внимание не только тактические, но и стратегические соображения. А стратегические соображения говорят, что у облачных сред большое будущее, так что если сегодня они, возможно, еще не созрели для немедленного применения крупным предприятием с большим количеством чувствительных данных, то их безусловно нужно изучать и, возможно, опробовать, имея в виду потенциальную значимость для развития информационных технологий (и не только их). Соображения перспективности и значимости облачных сервисов должны учитываться администраторами безопасности и руководством предприятия при выработке политики безопасности в отношении использования этих сервисов.

Вопросы к части VII

1. К какому виду нарушения безопасности относится глушение радиосигнала в беспроводных сетях:
   а) нарушение конфиденциальности;
   б) нарушение целостности;
   в) нарушение доступности;
   г) нарушение аутентичности.
2. Используя приведенный далее список терминов, вставьте пропущенные слова в следующее предложение: «Пользователи получают ... к ресурсам ИС в результате ..., однако прежде им необходимо успешно пройти ... и ...». (Аутентификация, авторизация, идентификация, права доступа.)
3. Вставьте пропущенные слова из списка: «Стало известно, что в интернете уже появились ..., направленные на использование ... новой версии браузера. Реализация данной ... может привести к ..., которая нанесет ... нашему предприятию». (Атака, уязвимость, эксплойт, ущерб, угроза).
4. Приведите примеры ситуаций, при которых обеспечивается конфиденциальность, но не гарантируется целостность данных.
5. Как называется свойство, которое характеризует систему, в которой документ всегда имеет достоверную информацию о его источнике (авторе)?
6. Сколько секретных ключей требуется для переписки 50 человек «каждый с каждым», использующих алгоритм DES?
7. Что определяет криптостойкость алгоритма шифрования:
   а) секретность алгоритма;
   б) секретность ключа;
   в) сложность алгоритма.
8. Можно ли передать секретный ключ по открытому каналу, не используя шифрование?
9. В главе 30 рассмотрен пример архитектуры сети с защитой периметра и разделением внутренних зон. Предложите альтернативный вариант.
10. Какие из перечисленных ниже свойств образуют триаду безопасности CIA:
    а) конфиденциальность;
    б) неотказуемость;
    в) аутентичность;
    г) владение;
    д) целостность;
    е) полезность;
    ж) захват привилегий;
    з) доступность.

11. Укажите, какие из перечисленных атак являются активными (выберите вариант ответа):
    а) прослушивание сетевого трафика;
    б) спуфинг;
    в) социальный инжиниринг;
    г) атака «отказ в обслуживании»;
    д) внедрение вируса.
12. Укажите, какие из следующих утверждений являются ошибочными:
    а) Нет смысла предусматривать подсистему аутентификации приложения, основанную на использовании многоразовых паролей, если многоразовые пароли уже используются и при логическом входе в систему.
    б) Надежная система авторизации может компенсировать недостаточную стойкость паролей пользователей, которые они использовали при входе.
    в) Стойкость системы защиты считается достаточной, если затраты на нее превысили запланированный уровень.
    г) Затраты на обеспечение безопасности информации, по крайней мере, не должны превышать величину потенциального ущерба от ее утраты.
    д) Иногда стойкость системы обеспечения безопасности считают достаточной, если стоимость ее преодоления злоумышленниками превосходит стоимость полученной ими выгоды.
13. Что из перечисленного может быть секретным ключом криптосистемы (выберите вариант ответа):
    а) геометрическая фигура;
    б) сборник детских сказок;
    в) картина;
    г) число 5.
14. В каких из перечисленных ниже средствах обеспечения безопасности используется шифрование (выберите вариант ответа):
    а) аутентификация;
    б) авторизация;
    в) протокол NAT;
    г) защищенный канал;
    д) сетевой экран прикладного уровня;
    е) фильтрующий маршрутизатор;
    ж) цифровая подпись.
15. Какую роль в обеспечении безопасности компьютерной сети играет протокол NAT?
16. Вставьте пропущенные слова (открытый или закрытый) в следующее предложение: «Сертификат может быть представлен в форме, состоящей из трех частей: во-первых, ... части; во-вторых, той же информации, зашифрованной ... ключом сертифицирующей организации, в-третьих, части, представляющей собой первые две части, зашифрованные... ключом владельца».

17. Установите в таблице правильное соответствие следующих терминовнарусском и английском языке.

ущерб	уязвимость	вредоносное ПО	доступность	целостность	подмена содержимого пакета	распределенная атака	«отказ в обслуживании»
Denial of Service	availability	integrity	vulnerability	malware	impact	spoofing	DDoS

18. Укажите, какие из нижеперечисленных свойств определяют одностороннюю функцию Y(x) (выберитевариаитответа):
    а) Y(x) чрезвычайно сложно вычислить для любого входного значения x;
    б) Y(x) просто вычисляется для любого входного значения x;
    в) аргумент x легко вычислить по известному Y;
    г) вычисление x по известному Y чрезвычайнозатруднено;
    д) вычисление x по известному Y абсолютно невозможно;
    е) значения функции от близких значений аргументов не должны значительно отличаться.
19. Поясните назначение алгоритма Диффи —Хеллмана (выберите вариант ответа):
    а) решает проблему передачи секретного ключа по открытому каналу;
    б) это то же самое, что и алгоритм RSA;
    в) смягчает проблему масштабируемости распределения ключей.
20. Сколько ключей требуется для секретной переписки 50 человек «каждый с каждым», использующих алгоритм RSA?
21. Какие из следующих утверждений правильные (выберите вариантответа):
    а) открытый ключ нужнозащищать от подглядывания;
    б) открытый ключ нужно защищать от подмены;
    в) теоретически возможно зашифровать текст одним ключом, а расшифроватьдругим, который абсолютно никак не связан с первым;
    г) алгоритм RSA основан на использовании функции разложения произведения большого числа на сомножители.
22. В чем состоит ручная синхронизация паролей? Варианты ответов:
    а) администратор синхронизирует значения БД паролей на разных серверах сети;
    б) ручная процедура приведения всех паролей пользователя к единому значению;
    в) использование одного и того же пароля для доступа к разным по степени защищенности сервисам.
23. Поясните, что представляют собой аутентификаторы из разряда «что-то знаю» (выберите вариант ответа):
    а) многоразовый пароль;
    б) одноразовый пароль, сгенерированный устройством;

в) преобразование пользователем в соответствии с некоторым правилом символов, выводимых системой на экран, и использование их в качестве пароля;

г) информация о месте и времени встречи.

24. Отметьте недостатки аутентификации с использованием многоразовых паролей (выберите вариант ответа):

а) возможность разгадывания пароля, если он слишком короткий;

б) необходимость передачи пароля по сети в открытом виде;

в) возможность подслушивания, подглядывания, «выманивания» пароля;

г) ручная синхронизация;

д) трудность запоминания надежных паролей;

е) сложность реализации.

25. Укажите, что из перечисленного содержится в сертификате (выберите вариант ответа):

а) информация о владельце сертификата;

б) открытый ключ владельца сертификата;

в) закрытый ключ владельца сертификата;

г) открытый ключ сертифицирующей организации;

д) закрытый ключ сертифицирующей организации;

е) информация о сертифицирующем центре, выпустившем данный сертификат.

26. Поясните, как убедиться в подлинности сертификата (выберите вариант ответа):

а) его надо расшифровать с помощью открытого ключа владельца сертификата и сравнить с открытой информацией сообщения;

б) его надо расшифровать с помощью закрытого ключа владельца сертификата и сравнить с открытой информацией сообщения;

в) его надо расшифровать с помощью открытого ключа сертифицирующей организации;

г) его надо послать для проверки в сертифицирующий центр.

27. Основная идея процедуры единого логического входа состоит в том, что (выберите вариант ответа):

а) пользователь выполняет логический вход в сеть только один раз при поступлении на работу, все остальное время система выполняет только его авторизацию;

б) все пользователи выполняют процедуру логического входа с одного и того же компьютера;

в) пользователь выполняет логический вход в сеть только один раз, затем результат этой аутентификации используется другими серверами или приложениями.

28. Укажите цели использования электронной подписи (выберите вариант ответа):

а) доказательство целостности сообщения;

б) обеспечение конфиденциальности сообщения;

в) доказательство авторства сообщения.

29. Поясните, что из перечисленного характеризует дискреционный метод управления доступом, а что — мандатный (выберите вариант ответа):

а) описание прав доступа дается в виде списков ACL;

б) право назначать права на доступ к объектам делегируются отдельным пользователям — владельцам объектов;

в) данная система управления доступом страдает от невозможности гарантированно проводить общую политику;

г) права доступа отдельного пользователя описываются ACE;

д) решение о предоставлении права доступа принимается операционной системой на основе правила;

е) субъект может по своему усмотрению передать часть своих полномочий другим субъектам;

ж) данная система управления доступом позволяет гарантированно проводить общую политику.

30. Основными функциями файервола являются (выберите вариант ответа):

а) аутентификация;

б) авторизация;

в) аудит;

г) фильтрация трафика.

31. Как может выглядеть запись в расширенном списке доступа, запрещающая передачу сообщений ICMP «Перенаправление маршрута» от любого хоста к хостам подсети 145.8.146.0/24?

32. Что означает данный список доступа:

access-list 2 permit 93.8.25.2 0.0.0.0 access-list 2 deny 93.8.25.0 0.0.0.255?

33. Какие из следующих утверждений о технологии NAT справедливы? Варианты ответов:

а) устройство, поддерживающее NAT, заменяет внешние IP-адреса пакетов, которые использовались при маршрутизации пакета через интернет, внутренними;

б) устройство, поддерживающее NAT, фильтрует входящий трафик, отбрасывая все пакеты с частными IP-адресами назначения;

в) причина обращения к технологии NAT — дефицит адресов IPv4;

г) причина обращения к технологии NAT — скрытие адресов хостов для повышения безопасности сети.

34. Какова должна быть защита общедоступного сервера (компьютера-бастиона)? Варианты ответов:

а) поскольку доступ к этому компьютеру открыт, его помещают в демилитаризованную зону, поэтому он не требует защиты;

б) уровень защиты компьютера-бастиона такой же, как у серверов внутренней сети;

в) компьютер-бастион должен быть особенно тщательно защищен от внешних пользователей;

г) правила, определенные для компьютера-бастиона по доступу к ресурсам внутренней сети, должны быть более строгими, чем правила, регламентирующие доступ к нему внешних пользователей.

35. В чем состоят главные отличия системы обнаружения вторжений (IDS) от файерволов с запоминанием состояния сеанса? Варианты ответов:
    а) IDS не блокирует подозрительный трафик;
    б) IDS анализирует не только события, связанные с прохождением трафика, но и другие подозрительные события в сети;
    в) IDS не выполняет журнализацию событий;
    г) в отличие от файерволов, системы IDS всегда являются исключительно программными.
36. Справедливо ли следующее утверждение: «Прокси-сервер всегда работает в режиме запоминания состояния сеанса»?
37. Какие функции системы безопасности из перечисленных направлены на обеспечение подотчетности (выберите вариант ответа):
    а) авторизация;
    б) аудит;
    в) протоколирование событий;
    г) мониторинг;
    д) журнализация событий.
38. В IDS для обнаружения вторжений применяются несколько типов правил:
    а) правила, основанные на сигнатуре атаки;
    б) правила доступа на основе меток безопасности объекта и субъекта;
    в) правила, основанные на анализе протоколов;
    г) правила, основанные на статистических аномалиях трафика.
39. В чем состоит главная уязвимость протокола IP? Варианты ответов:
    а) заголовок IP-пакета переносит адреса источника и назначения в незашифрованном виде;
    б) он использует широковещательные адреса назначения;
    в) он позволяет каждому узлу интернета взаимодействовать с каждым без предварительного установления соединения;
    г) он поддерживает фрагментацию пакетов.
40. С помощью протокола ICMP злоумышленник может (выберите вариант ответа):
    а) определить, что некоторый хост находится в работоспособном состоянии;
    б) организовать DoS-атаку;
    в) перенаправить маршрут;
    г) узнать, через какие промежуточные маршрутизаторы проходит маршрут до некоторого конечного узла.
41. С какой целью злоумышленник должен подавить отправку ACK-сегментов на атакуемый сервер в ходе атаки SYN Flood? Варианты ответов:
    а) чтобы скрыть свой IP-адрес;
    б) чтобы открытые соединения оставались незавершенными;
    в) чтобы закрыть открытые соединения.

42. Чем атака DNS-спуфинга отличается от атаки отравления DNS-кэша? Варианты ответов:
    а) ничем, это разные названия одной и той же атаки;
    б) в результате атаки отравления DNS-кэша DNS-сервер терпит крах, в то время как атака DNS-спуфинга к краху сервера не приводит;
    в) в атаке DNS-спуфинга ложный ответ предается клиенту, а в атаке отравления DNS-кэша — DNS-серверу;
    г) в атаке DNS-спуфинга ложный ответ передается DNS-серверу, а в атаке отравления DNS-кэша — клиенту.
43. Каким образом можно «подделать» маршрутное объявление BGP, которое вы передаете вашему соседу, если ваша автономная система является транзитной для этого маршрута, а вы хотите, чтобы сосед не использовал этот маршрут для передачи трафика (выберите вариант ответа):
    а) добавить в объявление номер автономной системы вашего соседа;
    б) выбросить из последовательности определенную автономную систему;
    в) добавить номер своей автономной системы несколько раз;
    г) заменить своими адрес сети и номер исходной автономной системы.
44. С какой целью в семействе протоколов IPSec функции обеспечения целостности дублируются в двух протоколах — AH и ESP?
45. Какую цель обычно преследует злоумышленник, используя эффект переполнения стека? Варианты ответов:
    а) испортить локальные переменные функции ядра ОС;
    б) поместить в стек вредоносный код и передать ему управление;
    в) исчерпать оперативную память компьютера.
46. Антивирусная программа, работающая по методу сигнатур, может (выберите вариант ответа):
    а) выявлять только известные вредоносные коды;
    б) выявлять только вирусы, но не черви;
    в) выявлять «троянские кони»;
    г) выявлять только новые виды вредоносных кодов.
47. Могут ли куки, переданные веб-сервером вашему веб-браузеру, заразить ваш компьютер вирусом?
48. С какой целью веб-сервер передает куки веб-браузеру клиента? Варианты ответов:
    а) для создания динамических веб-страниц;
    б) для сохранения состояния сеанса пользователя;
    в) для повышения защищенности сеанса.
49. Какое утверждение относительно протокола HTTPS является правильным? Варианты ответов:
    а) протокол HTTPS использует защищенный канал TSL для предотвращения атак на сеанс между веб-браузером и веб-сервером;

б) протокол HTTPS не является протоколом в строгом смысле этого термина;

в) протокол HTTPS использует цифровые сертификаты веб-браузера и веб-сервера для образования защищенного канала.

50. Что делает облачные вычисления более безопасными, чем традиционные? Варианты ответов:

а) высокая квалификация специалистов предприятий, предоставляющих облачные сервисы;

б) высокое качество вычислительной платформы;

в) способность поглощать трафик DDoS;

г) централизация вычислительных ресурсов.

Ответы

Ответы на вопросы к части I

1. –
2. Интернет можно охарактеризовать, например, следующим образом: глобальная публичная сеть с коммутацией пакетов, имеющая смешанную топологию, использующая стек протоколов TCP/IP, включающая магистральные сети, сети доступа и агрегирования трафика. Интернет состоит из сетей операторов связи и их клиентов.
3. Вычислительные ресурсы многотерминальных систем централизованы, а в компьютерной сети они распределены.
4. –
5. Иерархическая звезда.
6. Варианты а), б), в), г).
7. Варианты а), б), г).
8. Варианты а), б), г).
9. Варианты а), б), в), г).
10. Практически невозможно.
11. Вариант б).
12. То, в котором фиксируется маршрут.
13. Варианты а), г).
14. Невозможность динамического перераспределения пропускной способности физического канала между абонентами, возможность отказа в соединении, необходимость предварительного установления соединения, неэффективность использования пропускной способности при передаче пульсирующего трафика.
15. Наличие очередей и связанный с этим случайный характер задержек.
16. Варианты в), г).
17. –
18. Пропускная способность должна быть кратной элементарному каналу.
19. Договоренность о подтверждении отправлений делает этот обмен почтой похожим на установление логического соединения.
20. Варианты а), б), в), д).
21. Варианты а), б), г).
22. –

23. –
24. –
25. Сетевые службы в основном относятся к прикладному уровню. Модель OSI не включает пользовательские приложения.
26. Да, конечно; например, стек TCP/IP состоит из 4 уровней.
27. Нет, так как протокол не обязательно является стандартным.
28. Да.
29. Нельзя.
30. –
31. Стандартное отклонение: 17,2, коэффициент вариации: 0, 27.
32. Скорость потока может быть уменьшена путем отбрасывания пакетов.
33. Скорость передачи отдельного пакета равна пропускной способности линии — 100 Гбит/с.
34. Дискретизация по времени соответствует частоте квантования амплитуды звуковых колебаний 1/25 мкс, или 40 000 Гц. Для кодирования 1024 градаций звука требуется 10 двоичных разрядов. Отсюда необходимая пропускная способность для передачи оцифрованного таким образом голоса равна 40 000 × 10 = 400·Кбит/с.
35. Взвешенные очереди.
36. –
37. –
38. Да, может.
39. Вытеснение низкоприоритетного трафика.
40. При активной схеме измерений на результат могут повлиять измерительные пакеты, при пассивной схеме — недостаточная синхронизация.
41. Маршрут.
42. Буфер в пакетных сетях необходим всегда.
43. 5 Кбит, 20 Кбит и 75 Кбит бит выбирается из каждой очереди за один цикл.
44. Варианты: б), г).
45. –
46. –
47. Нет, не верно, так как пакеты наивысшего приоритета могут образовать собственную очередь.

Ответы к части II

1. Да.
2. Вариант в).
3. –

4. Вариант в).
5. 193,5 ТГц.
6. Да.
7. Варианты а), в) и г).
8. Вариант б).
9. –
10. Нет.
11. Влияние электромагнитного поля, создаваемого передатчиками, на соседние провода кабеля, к которым подключены входы приемников.
12. С большим по абсолютной величине значением NEXT (которое всегда отрицательно).
13. Вариант б).
14. 258,5 Мбит/с.
15. Да.
16. 400 бит/с.
17. Нет.
18. Вариант в).
19. 6
20. –
21. Вариант б).
22. –
23. Варианты а) и б).
24. Варианты б) и в).
25. Асинхронный.
26. 25 МГц и 75 МГц.
27. В 3 раза.
28. Варианты а) и в).
29. Варианты а) и в).
30. Вариант б).
31. Варианты а) и б).
32. Вариант б).
33. Вариант б).
34. –
35. Вариант в).
36. Варианты а), б) и в).
37. –
38. Вариант в).

39. –
40. Да.
41. Вариант б).
42. Варианты а), в).
43. –
44. Кадры SDH помещаются в кадры OTN как поток байтов, границы кадров игнорируются.
45. Варианты а), в).
46. Варианты б), в).

Ответы к части III

1. Верно.
2. Неверно.
3. Индивидуальный локальный.
4. Вариант а).
5. Вариант б).
6. Вариант а).
7. Вариант б).
8. Варианты б), в) и г).
9. Неверно.
10. Вариант б).
11. Да.
12. Варианты в) и г).
13. Варианты а), в) и г).
14. Назначив ему наибольшее значение идентификатора.
15. Нет.
16. Вариант а), в) и г).
17. deny f8:f2:1e:0c:3a:b8 ac:1f:6b:64:c7:d6.
    permit any any.
18. Нет.
19. Назначить ему отличное от группы значение административного ключа.
20. Вариант а).
21. Удаляет поле тега.
22. Варианты б) и в).
23. Варианты а) и в).
24. Нет.

Ответы к части IV

1. Синтаксически ошибочные адреса (6) 17.213.355.205 и (14) 12.145.3.12.5.
2. Класс A — (1) 123.1. 223.5, (4) 10.124.251.252, 5) 125.24.255.255, (8) 127.0.23.55, зарезервирован, (9) 1.0.0.13, (10) 124.1.1.1, (13) 13.13.13.13; класс B — (7) 179.12.255.255, (15) 172.17. 1.4; класс C — (3) 194.87.45.255, (11) 192.13.16.255.
3. Широковещательными адресами класса C являются (3) 194.87.45.255 и (11) 192.13.16.255.
4. Следующие адреса не могут быть назначены в качестве индивидуального глобального адреса интерфейса: (2) 225.0.0.1 — групповой адрес, (3) 194.87.45.255, (7) 179.12.255.255 и (11) 192.13.16.255 — широковещательные адреса, (8) 127.0.23.55 — внутренний адрес стека протоколов или адрес обратной петли, (15) 172.17.1.4 — частный адрес, (12) 0.0.0.0 — неопределенный адрес.
5. Номер подсети 108.5.18.160. Для нумерации интерфейсов в данной сети может быть использовано 4 бита, то есть 16 значений. Так как двоичные значения, состоящие из одних нулей и одних единиц, зарезервированы, то в сети не может быть более 14 узлов.
6. 64 подсети, маска /30.
7. Вариант г).
8. Между DNS-именами и IP-адресами в общем случае нет связи, поэтому ничего определенного сказать нельзя.
9. Вариант в).
10. Да, для двухточечных соединений стандарт разрешает использовать адреса интерфейсов 0 и 1.
11. Варианты а), б), в).
12. –
13. Варианты б), д).
14. Варианты а), б), в), г).
15. Нет.
16. Вариант г).
17. Вариант а).
18. Вариант б).
19. Поскольку ARP-таблица строится для каждого интерфейса, то число таблиц для каждого из этих устройств равно количеству их сетевых интерфейсов с назначенными IP-адресами.
20. 20 адресов (при условии, что в сети установлен DHCP-сервер).
21. Вариант а).
22. Для агрегирования трафика.
23. Варианты а), б), г).

24. –
25. Вариант в).
26. Чем больше количество совпавших двоичных разрядов при сравнении адреса назначения из пакета и адреса из записи таблицы маршрутизации, тем более специфическим является соответствующий маршрут. Предпочтение отдается наиболее специфическому маршруту.
27. Вариант б).
28. Нет. Для правильной маршрутизации пакетов в сети с использованием масок достаточно того, что маски передаются протоколами маршрутизации RIP-2, OSPF или устанавливаются вручную для каждой записи таблицы маршрутизации.
29. –
30. –
31. Запись 0.0.0.0 с маской 0.0.0.0 определяет наименее специфический маршрут, который выбирается только при отсутствии совпадений с другими записями таблицы.
32. –
33. Вариант а).
34. Вариант б).
35. Да, если оно для обеспечения надежности возьмет на себя функции, подобные функциям TCP, например квитирование, тайм-аут и т. п.
36. Объем полученных данных — 145 005 байт.
37. Вариант г).
38. –
39. а) Приемник устанавливает окно приема, б) отправитель — окно перегрузки.
40. Варианты а), в).
41. Варианты а), б), в), е).
42. Варианты а), в), г).
43. Варианты а), б), в), д).
44. Вариант в).
45. 00-80-48-EB-7E-60.
46. Варианты в), г).
47. Варианты а), б), г), д).
48. Так как маловероятно, что в локальной сети найдется несколько узлов, у которых совпадают три последних байта MAC-адреса.
49. –
50. Варианты а), б), г).
51. Вариант в).
52. –
53. Используя алгоритм Happy Eyeballs.
54. –

Ответы к части V

1. Варианты б) и в).
2. Вариант б).
3. Неверно.
4. Варианты б) и в).
5. Вариант б).
6. Вариант а).
7. Неверно.
8.

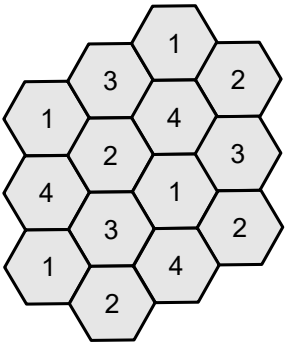
Путь к объекту	/moblle/web/versions.shtml
DNS-имя сервера	www.bbc.co.uk
URL-имя	http://www.bbc.co.uk/mobile/web/versions.shtml
Тип протокола доступа	http://

9. Варианты б), в), е).
10. Варианты а) и г).
11. Варианты б) и в).
12. Верно.
13. Вариант б).
14. Варианты б) и в).
15. Варианты а) и г).
16. Вариант б).
17. Вариант б).
18. Варианты а) и д).
19. Варианты а), в) и г).
20. Варианты а), б) и в).
21. Нет.
22. Вариант а).
23. Да.
24. Вариант в).
25. Вариант в).
26. Вариант б).
27. Вариант б).
28. Варианты а), б) и в).
29. Да.

30. Да.
31. Да.
32. Вариант в).
33. Нет, существуют топологии, для которых это невозможно.
34. Варианты а) и б).
35. Варианты б) и в).

Ответы к части VI

1. –
2. Варианты в) и д).
3. Варианты а), б), в), г).
4. Варианты б) и г).
5. Варианты б), в), г), д).
6. Видимого света.
7. Эффекта дифракции.
8. За счет распределения энергии полезного сигнала в широком диапазоне частот, так что узкополосные помехи не оказывают существенного влияния на сигнал в целом.
9. Как отношение выходной мощности, излучаемой данной антенной в определенном направлении, к выходной мощности, излучаемой идеальной изотропной антенной в любом направлении, при условии, что на вход обеих антенн поступают равные по мощности сигналы.
10. $\lambda/2$.
11. «Пространственное мультиплексирование увеличивает скорость, а пространственное разнесение увеличивает надежность».
12. Быстрого.
13. За счет ортогональности кодов расширения, назначенных каналам.
14. 20 кГц.
15. Данные передаются медленнее, так как станция откладывает передачу кадра, считая, что среда занята, хотя она свободна.
16. В сетях Ethernet коллизии распознаются (Collision Detection), а в сетях Wi-Fi — коллизии предотвращаются (Collision Avoidance).
17. За счет отсчета времени с момента окончания передачи кадра.
18. Увеличение дальности связи достигается за счет уменьшения битовой скорости передачи данных в два раза, до 1 Мбит/с, и применения кодов FEC.
19. Вариант с повторным использованием четырех частот показан на рисунке. В этом варианте сигналы смежных сот интерферируют меньше, чем в варианте с тремя частотами.



20. Уменьшить размер соты.
21. Потому что локализация телефона происходит с точностью до области отслеживания, в которую входит несколько сот.
22. Варианты а) и г).
23. Два ресурсных блока.
24. Упрощается процедура перестройки туннелей в большинстве случаев.
25. SDN, NFV, технология виртуальных машин.
26. Преимущества: более высокая скорость передачи данных за счет более широкой полосы частотных подканалов и компактные антенны вследствие короткой длины волны. Недостатки: меньшее расстояние связи из-за большего затухания высокочастотного сигнала, а также высокие требования к точности наведения луча.
27. Варианты: в), д).
28. Путем шифрования номеров ресурсных блоков, предназначенных телефону, с помощью локального идентификатора телефона C-RNTI.
29. Варианты б), в) и д).
30. Варианты а), б) и г).
31. За счет механизма двойного подключения.
32. Нет.
33. Вариант а).
34. Потому что у агрегированного кадра имеется только один заголовок с одним адресом назначения станции.
35. Варианты а) и б).
36. Роуминг в сетях Wi-Fi требует полного повторного присоединения станции к сети, а при эстафетной передаче чаще всего требуется только упрощенная процедура присоединения (без аутентификации и создания туннелей GPT).
37. Варианты а), б), в) и г).
38. Варианты а), б), в).
39. Варианты а), б), в), г), е).
40. Варианты а), б), в), г), д).

41. Она обеспечивает низкую чувствительность к помехам, а следовательно, возможность передачи маломощных сигналов на дальнее расстояние.
42. –
43. Варианты а), б), г), е).
44.

	Ad Hoc	Примеры узлов			Степень мобильности		
		Дрон	Автомобиль	Рация	Низкая	Средняя	Высокая
MANET	+			+	+		
VANET	+		+			+	
FANET	+	+					+

45. –
46.

	Wi-Fi (802.11s в режиме Ad Hoc)	LoRaWAN	Wi-Fi (802.11 a/b/g/n/ac/ ax)	4G/5G	Спутниковая связь	Диапазон mmWave технологии 5G
Дрон-дрон	+	+				
Шлюз-шлюз			+	+	+	
Дрон инфраструктура		+	+		+	+

47. Варианты а), б), в), г).
48. Варианты б), в), г), д), е).
49. За счет последовательной нумерации маршрутных объявлений.
50. Варианты в), г), д), е).

Ответы к части VII

1. Вариант в).
2. «Пользователи получают права доступа к ресурсам ИС в результате авторизации, однако прежде им необходимо успешно пройти идентификацию и аутентификацию».
3. «Стало известно, что в интернете уже появились эксплойты, направленные на использование уязвимости новой версии браузера. Реализация данной угрозы может привести к атаке, которая нанесет ущерб нашему предприятию».

4. Передача зашифрованной информации по открытому каналу.
5. Аутентичность.
6. 1225.
7. Вариант б).
8. Да, с помощью алгоритма Диффи — Хеллмана.
9. В зависимости от решаемых задач некоторые зоны могут быть объединены, а другие — разделены.
10. Варианты а), д), з).
11. Варианты б), г), д).
12. Варианты а), б), в).
13. Все варианты — а), б), в), г).
14. Варианты а), б), г), д), ж).
15. Скрывает внутреннюю структуру сети.
16. «Сертификат может быть представлен в форме, состоящей из трех частей: во-первых, открытой части; во-вторых, той же информации, зашифрованной закрытым ключом сертифицирующей организации; в-третьих, части, представляющей собой первые две части, зашифрованные закрытым ключом владельца».
17.

ущерб	уязвимость	вредоносное ПО	доступность	целостность	подмена содержимого пакета	распределенная атака	отказ в «обслуживании»
impact	vulnerability	malware	availability	integrity	spoofing	DDoS	Denial of Service

18. Варианты б), г).
19. Варианты а), в).
20. 100, 50 открытых и 50 закрытых ключей.
21. Варианты б), г).
22. Вариант в).
23. Варианты а), в), г).
24. Варианты а), в), г), д).
25. Варианты а), б), е).
26. Сначала а), затем в).
27. Вариант в).
28. Варианты а), с).
29. Дискреционный — а), б), в), г), е), мандатный — д), ж).
30. Варианты в), г).
31. access-list 120 deny ICMP any 145.8.146.0 0.0.0.255 eq 5.

32. Разрешить прохождение через маршрутизатор пакетов хоста 93.8.25.2, запрещая передачу пакетов, отправляемых любым другим хостом подсети 93.8.25.0/24.
33. Варианты а), в), г).
34. Варианты в), г).
35. Варианты а), б).
36. Да.
37. Варианты б), в), г), д).
38. Варианты а), в), г).
39. Варианты в), г).
40. Варианты а), б), в), г).
41. Вариант б).
42. Вариант в).
43. Варианты а), в).
44. –
45. Вариант б).
46. Вариант а).
47. Нет.
48. Вариант б).
49. Варианты а), б), в).
50. Варианты а), б), в).

Рекомендуемая и использованная литература

Самую полную и достоверную информацию можно почерпнуть из стандартов: IETF RFC — о протоколах стека TCP/IP, ITU-T — о технологиях первичных сетей, IEEE 802 — о технологии Ethernet и Wi-Fi, 3GPP — о технологиях мобильных сотовых сетей.

1. *Стивен Браун.* Виртуальные частные сети. М.: Лори, 2001.
2. *Шринивас Вегешна.* Качество обслуживания в сетях IP. М.: Вильямс, 2003.
3. *Галатенко В.* Основы информационной безопасности. М.: Бином. Лаборатория знаний, 2008.
4. *Аннабел З. Додд.* Мир телекоммуникаций. Обзор технологий и отрасли. М.: ЗАО «Олимп-Бизнес», 2002.
5. *Кейт Дж. Джонс.* Анти-хакер. Средства защиты компьютерных сетей. Справочник профессионала. 2003.
6. *Оливер Ибе.* Сети и удаленный доступ. Протоколы, проблемы, решения. М.: ДМК Пресс, 2002.
7. *Дуглас Э. Камер.* Сети TCP/IP. Том 1. Принципы, протоколы и структура. М.: Вильямс, 2003.
8. *Кеннеди Кларк, Кевин Гамильтон.* Принципы коммутации в локальных сетях Cisco. 2003.
9. *Куроуз Дж., Росс* К. Компьютерные сети. Нисходящий подход. М.: Эксмо, 2016.
10. *Лапонина О. Р.* Основы сетевой безопасности: криптографические алгоритмы и протоколы взаимодействия: Курс лекций. М.: Бином. Лаборатория знаний, 2009.
11. *Одом Уэнделл.* Официальное руководство Cisco по подготовке к сертификационным экзаменам CCNA ICND2 200-105. Маршрутизация и коммутация. М.: Вильямс, 2018
12. *Олифер В. Г., Олифер Н. А.* Новые технологии и оборудование IP-сетей. СПб.: БХВ — Санкт-Петербург, 2000.
13. *Олифер В. Г., Олифер Н. А.* Сетевые операционные системы, 2-е изд. СПб.: Питер, 2008.
14. *Олифер В. Г., Олифер Н. А.* Основы компьютерных сетей. СПб.: Питер, 2009.
15. *Олифер В. Г., Олифер Н. А.* Безопасность компьютерных сетей. М.: Горячая Линия — Телеком, 2015.
16. *Фейт Сидни.* TCP/IP. Архитектура, протоколы, реализация. М.: Лори, 2000.
17. *Сергиенко А. Б.* Цифровая обработка сигналов. СПб.: Питер, 2002
18. *Сингх Саймон.* Книга шифров. Тайная история шифров и их расшифровки. М.: АСТ, Астрель, 2007.
19. *Слепов Н. Н.* Синхронные цифровые сети SDH. М.: Эко-Трендз, 1998.
20. *Трещиков В. Н., Листвин В. Н.* DWDM-системы, 5 изд. М.: Техносфера, 2024

21. *Ибрагимов Р. З., Фокин В. Г.* Проектирование современных оптических транспортных сетей связи: Учебное пособие. СПб.: Лань, 2023
22. *Тихвинский В. О., Терентьев С. В., Коваль В. А. Девяткин Е. Е.* Развитие сетей мобильной связи от 5G Advanced к 6G: проекты, технологии, архитектура. Издание 2-е, доп. Техносфера, 2024.
23. *Степутин А. Н., Николаев А. Д.* Мобильная связь на пути к 6G. В двух томах. Инфра-Инженерия, 2020.
24. *Петерсон Л., Дэви Б.* Компьютерные сети. Системный подход. М.: АСТ, 2026.
25. *Пролетарский А. В., Смирнова Е. В., Ромашкина Е. А.* Технология современных беспроводных сетей Wi-Fi. Издательство МГТУ им. Н. Э. Баумана, 2017.
26. *David Coleman.* Wi-Fi 6 & 6E For Dummies®, Extreme Networks Special Edition, John Wiley & Sons, Inc. 2023
27. *Марк Спортак, Френк Паппас и др.* Компьютерные сети и сетевые технологии: ТИД «ДС», 2002.
28. *Степутин А. Н., Николаев А. Д.* Мобильная связь на пути к 6G. Инфра-Инженерия, 2018.
29. *Стивенс Ричард.* Протоколы TCP/IP. Практическое руководство. СПб.: БХВ, 2003.
30. *Столлингс В.* Передача данных, 4-е изд. СПб.: Питер, 2004.
31. *Столлингс В.* Современные компьютерные сети, 2-е изд. СПб.: Питер, 2003.
32. *Столлингс В.* Беспроводные линии связи и сети. Диалектика-Вильямс, 2003.
33. *Таненбаум Э. С., Фимстер Н., Уэзеролл Д.* Компьютерные сети, 6-е изд. СПб.: Питер, 2025.
34. *Уолрэнд Дж.* Телекоммуникационные и компьютерные сети. Вводный курс. М.: Постмаркет, 2001.
35. *Халсалл Фред.* Передача данных, сети компьютеров и взаимосвязь открытых систем. М.: Радио и связь, 1995.
36. *Чирилло Джон.* Защита от хакеров. СПб.: Питер, 2002.
37. *Харрис Ш.* CISSP. Руководство для подготовки к экзамену. 2011.
38. *Щербо В. К.* Стандарты вычислительных сетей. Взаимосвязи сетей: Справочник. М.: Кудиц-образ, 2000.
39. *Jerome Henry, Binita Gupta, Brian Hart, Malcolm Smith.* Wi-Fi 7 In Depth: Your Guide to Mastering Wi-Fi 7, the IEEE 802.11be Protocol, and Their Deployment. Addison-Wesley Professional, 2024
40. *Sauter Martin.* From GSM to LTE-Advanced, John Wiley & Sons, Ltd, 2014.
41. *Parker Donn B.* Fighting Computer Crime: A New Framework for Protecting Information, 1998.
42. *Davies Josef.* Understanding IPv6, 3E, Pearson, 2012.
43. *Peltier Thomas R.* Information Security Risk Analysis, Third Edition, 2010.
44. *Kabiri Peyman.* Privacy Intrusion Detection and Response, IGI Global, 2011.

45. *Alireza Maleki, Ha H. Nguen, Ebrahim Bedeer, Robert Barton,* A Tutorial on Chirp Spread Spectrum Modulation for LoRaWAN: Basics and Key Advances, IEEE ComSoc, VOLUME 5, 2024, pp. 4578 – 4612.
46. *Fabio Busacca, Stefano Mangione, Sergio Palazzo, Francesco Restuccia, Ilenia Tinnirello.* SDR-LoRa, an open-source, full-fledged implementation of LoRa on Software-Defined-Radios: Design and potential exploitation. Computer Networks, Volume 241, March 2024.
47. *Theodore S. Rappaport, Robert W. Heath. Robert C. Daniels, James N. Murdock.* Millimeter Wave Wireless Communications. Pearson, 2022.
48. *Solomon Michael G.* Security Strategies in Windows Platforms and Applications – Jones & Bartlett Learning, 2010.
49. *Pfleeger Charles P., Lawrence Shari.* Security in Computing, Fourth Edition – Prentice Hall, 2006.
50. *Noonan Wes, Dubrawsky Ido.* Firewall Fundamentals – Cisco Press, 2006.
51. *Pieprzyk Josef, Hardjono Thomas, Seberry Jennifer.* Fundamentals of Computer Security – Springer, 2010.
52. *Cole Eric.* Network Security Bible, 2nd Edition – John Wiley & Sons, 2009.
53. *Kocher Paul, Jaffe Joshua, Jun Benjamin.* Introduction to Differential Power Analysis and Related Attacks, 1998.
54. *Y. Zeng, Yong, I. Guvenc, R. Zhang, G. Geraci, D. Matolak.* UAV Communications for 5G and Beyond. John Wiley&Sons, 2020.
55. *D. N. K. Jayakody, P. Muthuchidambaranathan, Rui Dinis, Stefan Panić.* Integration of Unmanned Aerial Vehicles in Wireless Communication and Networks. Springer, 2023.
56. *Lav Gupta, Raj Jain, Gabor Vaszkun.* Survey of Important Issues in UAV Communication Networks. IEEE Communications Surveys & Tutorials, Volume 18, Issue 2, 2016.
57. *Mirmojtaba Gharibi, Raouf Boutaba, Steven L. Waslander.* Internet of Drones. IEEE Access, 2016.
58. *Naser Hossein Motlagh, Tarik Taleb, Osama Arouk.* Low-Altitude Unmanned Aerial Vehicles-Based Internet of Things Services: Comprehensive Survey and Future Perspectives. IEEE Internet of Things Journal, vol. 3, No. 6, December 2016.

Список материалов на сайте книги www.olifer.co.uk

1. Алгоритм ведра маркеров
2. Протоколы IntServ и DiffServ. Поддержка QoS-маршрутизаторами
3. Скремблирование и компрессия данных
4. Технология PDH
5. Технология SDH
6. Форматы кадров Ethernet
7. Конструктивное исполнение коммутаторов
8. Агрегирование линий связи
9. Технология PBB
10. Перекрытие адресных пространств
11. Технологии Mobile IPv6 и Proxy Mobile IPv6
12. Структурирование адресного пространства группового вещания
13. Протоколы маршрутизации группового вещания
14. Междоменное групповое вещание
15. Функциональная схема и классификация маршрутизаторов. Операционная система Cisco IOS
16. Технологии OpenFlow и P4
17. Многослойное представление технологий и услуг глобальных сетей
18. Технология X.25
19. Технология Frame Relay
20. Технология ATM
21. Технология MPLS VPN третьего уровня
22. Пространственно-временно́е кодирование STC
23. Циклический префикс в OFDM
24. Сети ISDN
25. Технологии доступа: PPP, HDLC, ADSL, PON, модемы
26. Сравнение процедур создания туннелей в сетях LTE и 5G
27. Симметричный блочный алгоритм шифрования, использующий преобразования Фейстеля
28. Биометрическая аутентификация
29. Слабость метода DAC
30. Ролевое управление доступом. Иерархия ролей, отношение наследования

31. Формальные модели безопасного управления доступом
32. Аутентификация и авторизация в операционных системах
33. Классификация систем единого логического входа
34. Система Kerberos
35. Фильтрация маршрутных объявлений
36. Анализатор протоколов Tcpdump
37. Защита BGP
38. VPN на основе IPSec
39. Пример использования техники переполнения стека для организации атаки
40. Скрытые коммуникации и скрытые каналы
41. Проверка действительности сертификата веб-сервера
42. Соглашение об уровне обслуживания с провайдером облачных сервисов

Алфавитный указатель

4G 721
5G 761
  eMBB 764
  MIoT 764
  URLLC 764
  режим двойного подключения 770
  слайс 763
5G DC
  Dual Connectivity 770
5G mmWave 864
5G NR 864
  5G новое радио 768
5G NR Sidelink Mode 2 864
5G NTN 781, 864
10G Ethernet 318
100Base-FX 314
100Base-TX 315
802.15.4 823
1000Base-SX 317

A

AC 650
access list 955
ACL 815, 944
Adaptive Load Balancing 335
AdHoc-сеть
  определение 673
ADM 243
ADSL 563
AES 915
AH 1012
ALOHA 286, 288, 664, 832
AM 232
A-MPDU 800
A-MSDU 800
AODV 874
AP 784
Arcnet 34
ARP 66, 380, 604
ARPANET 31
ARP-запрос 381
ARP-кэш 383
AS 490
AWG 273

B

bandwidth
  определение 197
baseline 980
Basic NAT 971
beacon frames 788
BER 201
BGP 491, 492
BGP LS 634
BGPv4 492
Bluetooth 814
Bluetooth LE 818
bottleneck 60
BPDU 327
BPSK 227
bridge
  LAN 298
BSS 784

C

CA 935
capacity of a link 201
Carrier Ethernet 350

CBR 148
CCM 353
CD 294
CDMA 712
CGI 582
challenge 928
CHAP 929
CIDR 378, 423
Cisco 335, 955
cloud computing 590
CO 560
community string 603
cookies 1032
CoS 617
coverage zone 666
CRC 231
CS 292
CSMA/CA 794
CSMA/CD 292
CWDM 270

D

DCE 185
DCF 794
DDoS 903
DES 914
DHCP 393
DHCP-агент 395
DiffServ 437
DIFS 796
DL 688
DLCI 96
DM 502
DMZ 985
DNS 385
  зона 389
  корневые серверы 388, 391
  обратная зона 392
  сервер 389
DNSSEC 1005
DNS-спуфинг 1001
DoS 902
DownLink 688
DS
  распределительная система Wi-Fi 785
DSP 318
DSSS 711
DTE 185
DTN 880
DVA 474
DWDM 267
  частотный план 270
DXC 244

E

E-1 249
E-2 249
E-3 249
eBGP 495
EDFA 275
edge computing 558
EDR 818
EGP 476, 491
encapsulation 538
EPL 645
ESP 1012
ESS 806
Ethernet 34, 36, 104, 286
  40G/100G 319
  200G и 400G 320
  800G и 1,6T 323
  Carrier Ethernet 350
  агрегирование линий связи 335
EtherType 342
EVC 644
EVPL 646
Extended Service Set
  ESS 806

F

FANET 683, 847, 852, 854
Fast EtherChannel 335
Fast Ethernet 34, 314
FCS 292
FDDI 34
FDM 236
FEC 231, 615
FEXT 200
FHSS 708
FIFO
  обслуживание очередей 157
flow control 165
FLP 313
FM 232
FPGA 310
FQDN 387
Frame Relay 611
FTN 616
FTP 364
FTT
  Fast Fourier Transformation 189

G

GEO 774, 775
Get-request 602
GFP 259
Gigabit EtherChannel 335
Gigabit Ethernet 34, 315
GPS 776
GTP 731

H

HARQ 748
hash function 922
HTML 567
HTTP 364, 571
HTTPS 575, 1034

I

IAB 123
IANA 498
iBGP 495
IBM 34
ICANN 377
ICMP 365, 428
IDS 980
IEEE 802.1Q 340
IEEE 802.11 783
IETF 123
IFTT 189
IGMP 499
IGP 476, 491
IKE 1012
IMAP 589
IMS 753
IMSI 725
information rate 60
Intel 335
Internet 31
IntServ 437
IP 365
  DS-байт 398
  байт дифференцированного обслуживания 398
  время жизни 400
  идентификатор
    пакета 399
  контрольная сумма заголовка 400
  общая длина пакета 399
  протокол верхнего уровня 400
  смещение фрагмента 399
  тип сервиса 398
  флаг пакета 399
IPG 293
IPSec 1012
IPv6 513
IP-адрес 371

IP-маршрутизация 401
Iridium 777
IRTF 123
ISDN 749
IS-IS 123
ISOC 123
ISP 561
ITU-T 111
IX 562

J

jam-последовательность 294

K

Ka-диапазон 689
Ku-диапазон 689

L

LACP 338
LAG 336
LAN 33, 126
LCN 96
LDP 615, 622
LEO 774, 776
LER 614
LFIB 615
LHC 57
link budget 196
link-local адрес 515
LLC 289
LLC1 290
LLC2 290
LLC3 290
loopback address
  IPv6 516
LoRa 831
  chirp signal 831
  параметр SF (Spread Factor) 839
  чирп-сигнал 831, 836
LoRaWAN 831
LP 820
LSA 474, 484
LSP 615
LTE 721
  транспортный туннель 729
LTE Sidelink Mode 4 864

M

MA 292
MAC
  уровень 289
MAC-адрес 65
MAN 36, 126
MANET 683, 854
MEF 644
MEO 774, 776
MEP 353
mesh-узел 809
MIB 603
MII 312
MIMO 700
MIP 353
MME 724
Mobile IPv6 434
MPLS 611
MPLS L2VPN 646
MPLS L3VPN 646
MSTP 347
MTU 424
MultiLink Trunking 335
multipath signal propagation 694

N

NAP 562
NAPT 971
NAT 970
ND
  Neighbor Discovery 527

NEXT 200
NFV 509
NIC 46
NM 400
NMS 596
Nortel 335
Novell NetWare 34
NRZ 219
NSP 650

O

OADM 271, 274
OAM 352
ODU 255
OFDM 714
OFDMA 718, 744, 802
OPU 255
OSI 111
OSPF 484
  ABR 488
OTN 254
OTU 255
OUI 290
OWD 141
OXC 276

P

P4 508
PAN 813
PAN-координатор 824
PB 357
PCEP 633
PCM 234
PDH 249
PDU 114, 603
Permanent Virtual Circuit (PVC) 608
PGW 725
PHP 617
PHY 313
PIN 931
PIR 144
PKI 939
PON 563
POP 560
POP3 589
promiscuous mode 977
Proxy MobileIPv6 434
pseudowire 647
PVC 608

Q

QAM 227
QoS 151
  кондиционирование трафика 162
  резервирование
    ресурсов 168
  управление потоком 165
QPSK 227
QUIC 575

R

RARP 384
RB 745
RFC 123
RFC 1517 378
RFC 1518 378
RFC 1519 378
RFC 1520 378
RFC 1700 442
RFC 3232 442
RIP 476
ROADM 277
RP 502
RPF 503
RSA 921
RSTP 325, 330
RSVP 176
RSVP-TE 627

RTT 142
RU 803

S

SA 1013
SCO 815
SDH
  виртуальный контейнер 252
SDN 504
Set 603
SG 1014
SGW 725
SIFS 795
Simple Management Network Protocol (SNMP) 602
single-hop 669
SIP 749
SIR 143
SLA 132
slow start 467
SM 502
SMS 597
SMTP 364, 584
SN 1016
SNMP 602, 603
SNR
  signal-to-noise ratio 200
Spatial Diversity 702
Spatial Multiplexing 704
SPD 1019
SPF 631
SPI 1016
Spread Spectrum 708
SR-MPLS 628
SR-Policy 631
SR-путь 629
SSID 788
SSL 1011
STA 305, 325, 784
Starlink 778
STM-1 250
store-and-forward 91
STP 325
STP кабель 206
stream 573
streaming telemetry 605
SVC 608
Switched Virtual Circuit (SVC) 608

T

T-1 249
T-2 249
T-3 249
TCP 365
TCP-порт 442
TCP сегмент 445
TCP-сокет 443
TDM 237
TE 172, 626
telnet 364, 604
Thread
  технология 829
throughput 60
TLS 1011
TM 243
Token Bus 34
ToS 398
Traffic Engineering
  TE 173
TS 260
TTL 400, 425
tunneling 538
TXOP 800

U

UAV 843
UDP 365
UDP-дейтаграмма 444
UDP-порт 442

UDP-сокет 443
UE 725
UL 688
UNI 644
UpLink 688
URL 568, 569
UTP-кабель 207

V

VANET 683, 855
VBR 148
VCI 96
VLAN 338, 603
  линия
    доступа 344
  порт
    доступа 344
  транк 344
VNF 510
VPLS 646, 652
VPN 558, 638, 646
VPN второго уровня 644
VPN третьего уровня 644
VPWS 646, 649
vulnerability 897

W

WAN 30
WFQ 161
Wi-Fi 783
  ассоциация 787
  короткий межкадровый интервал SIFS 795
  маячковые кадры 788
  распределенный межкадровый интервал DIFS 796
  распределительная система 785
  ресурсная единица RU 803
  формат кадра 791
Wi-Fi 802.11s 809
Wi-Fi Extender 813
Wi-SUN FAN
  технология 829
WSN 683
WSS 278
WWW 567

X

X.25 31, 611

Y

Y.1731 356

Z

ZigBee
  технология 827

A

абонент 80
Абрамсон Норман 286
абсолютная мощность сигнала 195
автоматическое назначение
  динамических адресов 394
  статических адресов 394
автономная система 490
автопереговоры 313
авторизация 894
агент 602
агрегатный порт 244
агрегирование
  адресов 424, 517
  линий связи 333
  физических каналов 333
адаптер
  сетевой 46
адаптивная маршрутизация 473
администратор 411

адрес
IP-адрес 371
MAC-адрес 65
аппаратный 65, 370
виртуального интерфейса 401
выходного интерфейса 403
глобальный 369
групповой 64, 290, 373, 411
индивидуальный 290, 373
локальный 370
назначения 68, 403
неопределенный 374
обратной петли 375, 401
ограниченный 375
особого назначения 411
порта 409
произвольной рассылки 64
разрешение 66
сетевой 369, 371
символьный 64
следующего маршрутизатора 403, 409
уникальный 64
частный 377, 970
числовой 64
широковещательный 64, 290, 375, 411
адресация 64
иерархическая 65
плоская 65
адресная таблица 300, 301
адресное пространство 65
активное измерение 138
алгоритм
адаптивной маршрутизации 473
ведра маркеров 165
взвешенных очередей 160
Дейкстры 485
динамической маршрутизации 473
комбинированный 161
покрывающего дерева 305, 325
приоритетного обслуживания 158
прозрачного моста 300, 325
шифрования 921
альтернативный порт 331
амплитудная модуляция АМ 232
амплитудно-частотная характеристика АЧХ 192
анализ
надежности 596
производительности 596
анализатор протоколов 976
аналого-цифровой преобразователь АЦП 233
антенна 696
аппаратный адрес 65, 370
аппаратура передачи данных 185
арбитр
доступа 77
аренда
IP-адресов 394
каналов 558
асинхронное приложение 149, 150
атака
DDoS 903
активная 901
отказа в обслуживании 903
пассивная 902
понятие 898
распределенная 903
атакующий блок 1025
ATM 611
аудит 983
аутентикод 940
аутентификация 603
данных 893
пользователя 893
приложений 894
строгая 928
устройств 894
аутентичность 896

Б

база данных
  безопасных ассоциаций 1019
  политики безопасности 1019
база данных MIB 603
базовая сеть Wi-Fi 784
базовая станция 722
базовая трансляция сетевых адресов 971
базовый уровень характеристик сети 980
баланс нагрузки 93
бастион 986
безопасная ассоциация 1013
безопасность транспортных услуг 131
бесклассовая междоменная маршрутизация 378, 423
Беспилотный летательный аппарат
  БПЛА 843
беспроводная сеть 126
беспроводные технологии
  с низким энергопотреблением НЭ 820
БИС 32
битовая скорость
  переменная 148
  постоянная 148
битовые ошибки
  интенсивность
    BER 201
битовый интервал 314
бит синхронизации 249
бит-стаффинг 250
блок
  атакующий 1025
  данных оптического канала 255
  поиска целей 1025
  пользовательских данных оптического канала 255
  транспортный оптического канала 255
  управления 1027
  фиксации событий 1027
бод
  определение 204
большая интегральная схема 32
большой адронный коллайдер 57
бот 1030
ботнет 1031
бот-сеть 899
БПЛА 843
брандмауэр 958
браузер 569
брутфорс-атака 926
буфер 91
буферная память 91
быстрое преобразование Фурье 189
быстрый протокол покрывающего дерева 325
бюджет
  беспроводной линии связи 706

В

вариация задержки пакета 141
веб-документ 567
веб-клиент 569
веб-сервер 569
веб-страница 567
ведро маркеров 165
вектор атаки 1025
величина пульсации 144
вероятность отказа 145
взаимодействие
  межсетевое 368
взаимодействие открытых систем 111
взвешенная очередь 160
взвешенное обслуживание 160, 161
виртуализация сетевых функций 509
виртуальная локальная сеть 338
виртуальная сетевая функция 510
виртуальная частная линия Ethernet 646
виртуальная частная сеть 558, 638, 646
виртуальное соединение 644

виртуальный канал 96, 607
вирус 1027
внешний шлюз 491
внешний шлюзовой протокол 491
внешняя угроза 899
внутренний шлюзовой протокол 491
волновое мультиплексирование
  уплотненное 267
волновое мультиплексирование WDM 237
волновое сопротивление линии 199
волоконно-оптический кабель
  оптоволокно 208
волоконно-оптический усилитель 275
восходящее направление передачи 688
вредоносная программа 904
время
  буферизации 100
  жизни 411, 473, 480
  жизни записи 411
  коммутации пакета 133
  конвергенции 473
  наработки на отказ 145
  оборота 141, 142, 295
  ожидания пакета в очереди 133
  пакетизации 101
  передачи 100, 133
  распространения сигнала 100, 133
  реакции сети 141, 142
  сериализации 133
время задержки пакета
  определение 133
время сериализации 99
Всемирная паутина 567
вторжение 980
входная очередь 92
входной буфер 91, 102
выборка случайной величины 136
выделенный сервер 54
выравнивание
  заголовка пакета 400
выходная очередь 91

Г

гармоника 188
геокастинг 874
геостационарная 774
геостационарный 775
гиперссылка 568
гипертекстовая страница 567
гистограмма распределения 136
главное устройство 814
глобальная 776
глобальная метка потока 69
глобальная сеть 30, 126
глобальный адрес 369
городская сеть 126
группирование MAC-адресов 341
групповое вещание 497
групповой адрес 64, 290, 373, 411

Д

дайджест 922
дальность связи 668
двойной стек 536, 540
двоичный код 57
деградация системы 146
дейтаграмма 367
  понятие 114
дейтаграммная передача 92
дейтаграммная сеть 127
дейтаграммный протокол 366
декомпозиция
  иерархическая 107
  понятие 107
демилитаризованная зона 985
демультиплексирование 74, 441
демультиплексор 75
дерево 63
  разделяемое 502
  с вершиной в источнике 502
дешифрирование 912

джиттер 141
диаграмма направленности антенны 697
динамическая запись 302, 383
динамическая маршрутизация 473
динамическая страница 580
динамическая фрагментация 424
динамический номер порта 442
дискретизация
  по времени 82
  по значениям 82
дисперсия 268
дистанционно-векторные протоколы 474
дифракционная структура 273
дифракционная фазовая решетка 273
дифракция 269, 690
дифференцированное обслуживание 398, 437
длина пульсации 317
долговременные характеристики сети 132
доля потерянных пакетов 145
домашний агент 435
домен
  группового вещания 501
  имен 387
  коллизий 307
  широковещательного трафика 339
доменная система имен 386
доменная служба имен
  DNS 385
доменное имя 371, 386, 387
Доплера эффект 694, 777
доступ
  коллективный 292
  маркерный 288
  случайный 288
  терминальный 604
доступность 146, 895
драйвер
  переферийного устройства 47
  сетевой интерфейсной карты 46
древовидная топология 127
дрон 843
дуплексный канал 60
дуплексный режим
  коммутатора 307

E

единый логический вход 952
емкость канала связи 59
емкость линии связи
  capacity 201

З

заголовок
  аутентификации 1012
  вставка 617
  маршрутизации 524
  основной 523
  пакета 88
  фрагментации 524
задержка
  доставки пакета 137
  квантиль 138
  коэффициент вариации 138
  медиана 137
  процентиль 138
  среднее значение 137
  стандартное отклонение 137
закон Гроша 30
закрытый ключ 920
замораживание изменений 484
запись
  динамическая 302, 383
  статическая 302, 383
запрос
  понятие 84
затопление сети 302
затухание
  погонное 194
затухание оптического сигнала 210
затухание сигнала 193

защита
  линии 635
  пути 636
  узла 635
защищенный канал 1008
защищенный протокол IP 527
звезда 63
звездообразная топология 63, 127
звено
  link 182
зеркализация
  порта 140
зона покрытия
  определение 666

И

идентификатор
  виртуального канала 96
  запроса 433
  интерфейса 517
  организационно уникальный 290
  соединения 96
идентификатор абонента IMSI 725
иерархическая адресация 65
иерархическая декомпозиция 107
иерархическая звезда 63
иерархия скоростей 249
измерение
  активное 138
  пассивное 140
изохронное приложение 150
импульсно-кодовая модуляция 234, 249
импульсный способ кодирования 58
имя
  DNS-имя 371
  доменное 371, 387
  плоское 385
  символьное 371
индекс параметров безопасности 1016
индивидуальный адрес 290, 373
индивидуальный клиент 557
инжиниринг
  социальный 902
  трафика 172
  трафика в MPLS 626
инкапсуляция 538
интегрируемость сети 148
интенсивность
  отказов 145
интерактивное приложение 150
интервал
  битовый 314
  межпакетный 293
  отсрочки 314
интернет 37
интерфейс
  входной 72
  выходной 72
  коммутатора 72
  логический 46
  межуровневый 108
  независимый от среды 312
  понятие 45
  сетевой 64
  услуг 108
  физический 46
  шлюзовой 582
интерфейсная карта 47
интерференция 268
информационные параметры сигнала 203
информационные услуги 556
информационный поток 68, 367
инфраструктура с открытыми ключами 939
инфраструктурная сеть 672
ионосферные волны 692
истечение времени жизни маршрута 480

К

кабель 46
кабельные линии 184

кадр 367
  помеченный 343
  понятие 114
  продвижение 301
канал
  channel 182
  виртуальный 96
  дуплексный 60
  полудуплексный 60
  понятие 81
  присоединения 650
  связи 46
  симплексный 61
  составной 83
  спектральный 267
  элементарный 81, 82
канал тональной частоты 225
канальный уровень 112
качество обслуживания
  методы обеспечения 151
  понятие 130
квадратурный модулятор 229
квантиль 138
квитанция 59
квитирование 451
КВК 608
Керкхоффса правило 913
класс
  IP-адресов 373
  адресов 372, 374
  транспортного сервиса 113
  услуги 617
  эквивалентности продвижения 615
классификация
  компьютерных сетей 125
  критерии 125
  трафика 158
кластер дронов 854
клиент
  индивидуальный 557
  корпоративный 557, 558
  массовый 557
  понятие 51
  почтовый 582
клиентская операционная система 54
ключ
  закрытый 920
  открытый 913, 919
  секретный 912, 931
коаксиальный кабель 208
код
  4B/5B 222, 313
  AMI
    биполярное кодирование 220
  без возвращения к нулю
    NRZ 219
  биполярный импульсный 221
  двоичный 57
  избыточный 222
  манчестерский 221
  потенциальный с инверсией при единице
    NRZI 221
  Рида — Соломона 257
кодек 234
кодирование 203
  импульсный способ 58
  понятие 57
  потенциальный способ 58
коды Хемминга 232
коллективный доступ 292
коллизия 293
  обнаружение 294
  распознавание 295
кольцевая топология 63, 127
кольцо 63
  SDH 251
  плоское 251
комбинированное обслуживание 161
коммуникационное облако 51
коммуникационные услуги 556

коммутатор
3-го уровня 340
корневой 326
неблокирующий 308
пакетный 91
понятие 72, 73
фотонный 276
коммутатор Ethernet 306
коммутационная сеть 73
коммутация
интерфейсов 72
каналов 80
многопротокольная 611
пакетов 80, 88
по меткам 615
понятие 66, 73
коммутируемый виртуальный канал (КВК) 608
коммутирующий блок 92
коммутирующий по меткам маршрутизатор 614
компьютер-бастион 986
компьютерная сеть 27, 49
компьютерный трафик 87
конвейерная передача 572
конвергенция 473
конвергенция телекоммуникационных и компьютерных сетей 37
кондиционирование трафика 152
конечная точка обслуживания 353
контент 960
контроллер 47
контроль
допуска в сеть 169
расходования ресурсов 131
контрольная последовательность кадра 292
контрольная сумма 59, 106
пакета 88
контрольная сумма FCS 230
контроль по паритету 230
конфигурационный параметр 393
конфигурирование 393
конфиденциальность 895
концевик 88
концентратор 297
понятие 63
корневой коммутатор 326
корпоративная сеть 128
корпоративный клиент 557, 558
коэффициент
вариации 138
пульсации трафика 144
коэффициент загрузки
образование очередей 154
краткое доменное имя 387
краткосрочные характеристики сети 132
кратчайший маршрут 172
криптосистема
асимметричная 919
понятие 912
раскрытие 912
криптостойкость 913
критерий
выбора маршрута 399
классификации 125
кросс-коннектор
цифровой 244
куки 1032

Л

лавинная маршрутизация 471
линия связи 46, 58, 182
логическая топология
беспроводной сети 668
логический интерфейс 46
логическое соединение 94
локализация адресов 424
локальная метка потока 69
локальная сеть 33, 126
локальная таблица коммутации 85
локальное приложение 54

локальный адрес 370
локальный оператор 559
локальный поставщик услуг 562
локальный признак потока 85
локальный способ назначения адреса 290
лямбда 267

M

магистральная сеть 128
магистральный поставщик услуг 562
максимальная скорость передачи 167
маловысотная 774
манипуляция
  амплитудная
    ASK 225
  фазовая
    FSK 225
  частотная
    FSK 225
маркерный доступ 288
маршрут 96
  временный 411
  кратчайший 172
  понятие 66
  постоянный 411
  по умолчанию 404
  специфический 404, 411
  статический 411
маршрутизатор 73
  коммутирующий по меткам 614
  пограничный 614
  по умолчанию 404
  программный 408
маршрутизация 73
  адаптивная 473
  динамическая 473
  лавинная 471
  лавинная или эпидемическая 881
  от источника 472
  статическая 473
MAC
  адрес 289
маска 372, 375, 416
  двоичная запись 372
  понятие 372
Массовый интернет вещей 685
массовый клиент 557
масштабируемость сети 147, 370
медленный старт 467
межпакетный интервал 293
межсетевое взаимодействие 368
межсетевой протокол 365
межсетевой экран 958
межсимвольная интерференция 694
межуровневый интерфейс 108
менеджер
  протоколов 536
метка
  потока 69, 96
Меткалф Роберт 287
метод
  инжиниринга трафика 152, 172, 173
  кондиционирования трафика 152
  контроля перегрузок 132
  маркерного доступа 288
  обратной связи 152
  предотвращения перегрузок 132
  простоя источника 452
  скользящего окна 453
  случайного доступа 288
метрика 326
  производительности сети 135
механизм
  управления перегрузкой 151
мини-компьютер 32
минимальная таблица маршрутизации 411
многолучевое замирание 694
многолучеое распространение сигнала 694
многомодовый кабель оптический 210
многопортовый повторитель 297

многопротокольная коммутация по меткам 611
многотерминальная операционная система 31
многотерминальная система разделения времени 29
многоуровневый подход 107
множественный протокол покрывающего дерева 347
мобильная телефония 662
мобильность
  определение 676
Мобильный IP 434, 870
модель
  взаимодействия открытых систем 111
модем 225
модемы 563
модуляторы 228
модуляция 58, 203
  LoRa 837
  импульсно-кодовая 249
мост
  локальной сети 298
  понятие 298
  провайдера 357
  прозрачный 300
мультимедийная IP-система 749
мультиплексирование 74, 442
  волновое 267
  временное TDM 237
  с разделением времени TDM 237
мультиплексор 75, 244
  первичной сети 246
мультиплексор протоколов 536
мультисервисная сеть 36
мэйнфрейм 28

Н

навигационная система 857
надежность транспортных услуг 131
назначение
  динамических адресов 394
  статических адресов 394
Найквист, соотношение 205
наложенная сеть 129
наложенные сети 183
национальный оператор 560
неблокирующий коммутатор 308
недогруженная сеть 152
недогруженный режим 175
не зависимый от среды интерфейс 312
неопределенный адрес 374
неполносвязная топология 62
неразборчивый режим 291, 300
несущая частота 292
несущий протокол 538
несущий сигнал 203
неумышленная угроза 899
неэкранированная витая пара 207
нисходящее направление передачи 688
номер
  версии протокола IP 398
  порта 442
  сети 371
  узла в сети 371

О

область сети 487
облачные вычисления 590
обнаружение коллизии 294
обновление триггерное 483
обработка ошибок 596
обратная петля 375
обратная связь 152
обслуживание
  взвешенное 160, 161
  дифференцированное 398
  комбинированное 161
  приоритетное 158
  справедливое 161

общая среда передачи данных 286
общая шина 63, 104
общий шлюзовой интерфейс 582
объединение подсетей 423
объявление
  о расстоянии 474
  о состоянии связей сети 484
ограниченная широковещательная рассылка 375
ограниченный широковещательный адрес 375
ограничитель начала кадра 292
одномодовый кабель оптический 210
одноразовый пароль 930
одноранговая операционная система 54
односторонняя задержка пакетов 141
односторонняя функция 916
односторонняя функция шифрования 922
одношаговая сеть 669
окна прозрачности
  оптоволокно 195
окно
  приема 461
  скользящее 453
окно перегрузки 466, 578
оконечное оборудование данных 185
оператор
  локальный 559
  национальный 560
  операторов 559
  региональный 560
  связи 555
  транснациональный 560
операционная система
  клиентская 54
  компьютера 52
  многотерминальная 31
  одноранговая 54
  серверная 54
  сетевая 31, 52
опорная мощность сигнала 196
оптическая транспортная сеть 254
оптический кросс-коннектор 276
оптоэлектронный кросс-коннектор 276
организационно уникальный идентификатор 290
оркестрирование 511
ортогональное частотное мультиплексирование 714
основной заголовок IPv6 523
особый IP-адрес 411
отказ
  в обслуживании 902, 903
  в установлении соединения 86
отказоустойчивость 146
открытая система 121
открытая спецификация 121
открытый ключ 913, 919, 920
относительная мощность сигнала 195
относительное доменное имя 387
относительный коэффициент использования 161
отражение 269
очередь
  взвешенная 160
  входная 92
  выходная 91
  повторной передачи 463
  приоритетная 158
  теория очередей 154

П

пакет 367
  понятие 88, 114, 368
пакетный коммутатор 91
пакетный метод коммутации 37
пакетный шлюз PGW 725
память
  буферная 91
параметры
  логического соединения 94
  получателя 524

пароль 925
многоразовый 925
одноразовый 925
одноразовый 930
пассивное измерение 140
ПВК 608
пейджинг 743
первичная сеть 129, 182, 241
перегрузка
контроль 132
предотвращение 132
признак 167
управление 151
передача
голоса 31
дейтаграммная 92
конвейерная 572
последовательная 572
с простоями 572
с установлением соединения 94
перекрестные наводки
на ближнем конце 200
на дальнем конце 200
перекрестные наводки на ближнем конце 200
переменная битовая скорость 148
периферийные (граничные) вычисления 558
персональная сеть 813
персональный компьютер 34
петля 304
пиковая скорость передачи данных 144
пикосеть 814
планирование
расходования ресурсов 131
сети 147
плезиохронная цифровая иерархия PDH 249
плоская адресация 65
плоское имя 385
плоское кольцо 251
повторитель
многопортовый 297
пограничный маршрутизатор 614
пограничный шлюзовой протокол 491, 492
поддомен 387
подпоток 68
подчиненное устройство 814
покрывающее дерево 305, 325
поле
данных 292
источника записи 411
контрольной последовательности кадра 292
политика маршрута 473
политика маршрутизации 493
полное доменное имя 387
полносвязная топология 62, 127
полностью оптический кросс-коннектор 277
полоса пропускания
определение 197
полудуплексный канал 60
полудуплексный режим
коммутатора 307
пользовательский слой 119
пользовательский фильтр 303, 332
поляризационная модовая дисперсия 211
помехи
внешние 199
внутренние 199
помехоустойчивость линии 200
помеченный кадр 343
порог чувствительности приемника 196
порт 46
TCP-порт 442
UDP-порт 442
агрегатный 244
альтернативный 331
приложения 442
резервный 331
трибутарный 244

порядковый номер запроса 433
последовательная передача 572
поставщик услуг
  интернета 561
  локальный 562
  магистральный 562
  региональный 562
постоянная битовая скорость 148
постоянный виртуальный канал (ПВК) 608
потенциальный способ кодирования 58
поток
  байтов 445, 573
  данных 68, 367
  информационный 68, 367
потоковая телеметрия 605
потоковый трафик 148
почтовый клиент 582
почтовый сервер 582
преамбула 292, 295
предложенная нагрузка 59, 133
предотвращение
  перегрузки 151
пре-кодирование 703
преобразование Фурье
  обратное 189
  прямое 189
префикс 423
  формата 515, 516
признак
  непосредственно подключенной сети 410
  перегрузки 167
прикладной уровень 364
  уровень
    прикладной 114
приложение
  асинхронное 149, 150
  изохронное 150
  интерактивное 150
  локальное 54
  сверхчувствительное к задержкам 150
  сетевое 56
  синхронное 150
  с потоковым трафиком 148
  с пульсирующим трафиком 148
  устойчивое к потере данных 150
  чувствительное к потере данных 150
приоритет
  пакета 398
  понятие 158
приоритетное обслуживание 158
проактивная маршрутизация 475
проблема
  распределения ключей 915
провайдер 561
провайдер услуг 555
проверка непрерывности соединения 353
проводная сеть 126
проводные воздушные линии 184
программа
  вредоносная 904
  троянская 1024
  шпионская 904
программируемые протокольно-независимые пакетные процессоры P4 508
программное обеспечение стека TCP/IP 411
программно-определяемые сети 504
программный маршрутизатор 408
продвижение
  по реверсивному пути 503
продвижение данных
  понятие 72
продвижение кадра 301
прозрачный мост 300
производительность
  транспортных услуг 131
прокси-сервер
  понятие 966
  прикладного уровня 968
прокси-сервер SIP 750
промежуточная точка обслуживания 353

пропускная способность 59, 201
простой источника 452
простой протокол
  передачи почты 364
простой протокол передачи почты 584
пространственное мультиплексирование 704
пространственное разнесение 702
протокол
  CFM 353
  IPSec 1012
  IPv6 513
  IS-IS 489
  OpenFlow 506
  OSPF 484
  Proxy-ARP 384
  RTP 753
  SIP 749
  TCP 441
  VTP (VLAN Trunking Protocol) 346
  верхнего уровня 400
  взаимодействия приложений 48
  двухточечной связи 929
  дейтаграммный 366
  динамического конфигурирования хостов DHCP 393
  доступа к электронной почте 589
  инкапсуляции 538
  как логический интерфейс 46
  маршрутизации 411, 499
    AODV 874
    HWMP 811
  маршрутной информации RIP 476
  межсетевой 365
  межсетевых управляющих сообщений 365, 428
  ориентированный 585
  пассажир 538
  передачи 364, 584
  покрывающего дерева 325, 347
  пользовательских дейтаграмм 365
  понятие 109
  почтового отделения 589
  разрешения адресов 66, 380
  распределения меток 615, 622
  резервирования ресурсов 176
  сигнальный 615
  состояния связей 474
  управление агрегированием линий связи LACP 337
  управления 365
  шлюзовой 491, 492
  эмуляции терминала 364
протокол канального уровня 121
протокол обнаружения соседей 527
протоколы
  вспомогательные 119
  служебные 119
протоколы вектора пути 474
протокольная единица данных 114, 327
профилирование 162
профиль Bluetooth 816
процедура
  разрешения имени 391
  установления соединения 84
процентиль 138
процессор
  цифрового сигнала 318
прямая видимость 692
прямая коррекция ошибок FEC 231
прямая связь с телефоном 781
псевдоканал 647
пул адресов 395
пульсация 317
пульсирующий трафик 87
путь
  коммутации по меткам 615

P

радиоканалы 184
разделение
  времени 75

на подсети 423
ресурсов 45
частотное 75
разделяемая радиосреда 665
разделяемая среда 76, 665
разделяемое дерево 502
размер окна 454
разрешение адреса 66
распознавание коллизий 295
распределенная атака 903
распределенное приложение 56
рассеивание сигнала 690
рассредоточенная сеть 815
рассылка
ограниченная 375
расширение 316
расширенный спектр 659
расширяемость сети 146
расщепление горизонта 483, 653
реактивная маршрутизация 475
реверсивный протокол разрешения адресов 384
регенератор сигнала 244
региональный оператор 560
региональный поставщик услуг 562
режим
дуплексный 307
неблокирующий 309
недогруженный 175
неразборчивый 291, 300
перераспределения 490
полудуплексный 307
пульсаций 316
терминального доступа 604
транспортный 1014
туннельный 1014
удаленного управления 604
режим перераспределения маршрутов 490
резервирование
пропускной способности 168
резервирование ресурсов 168
резервная связь 305
резервный порт 331
рекурсивная процедура разрешения имени 391
ресурсный блок LTE 745
ресурсы
контроль расходования 131
планирование расходования 131
разделение 45
ретрансляционный участок 402
решетчатые коды
коды решетчатые 228
риск
понятие 900
рой дронов 845
роуминг 742, 808
ручное назначение статических адресов 394

С

самосинхронизирующиеся коды 219
связной агент 395
связь
резервная 305
сеансовый уровень 114
сеанс связи 80
сегмент 326, 367
локальной сети 298
понятие 114
сегмент маршрута 628
сегментная маршрутизация 472
SR-MPLS 628
секретный ключ 912, 931
сервер
выделенный 54
имен 66
понятие 51
почтовый 582
сетевой 34
серверная операционная система 54

сервер-регистратор SIP 750
сервер управления мобильностью 724
сервисный шлюз SGW 725
сертификат 935
сетевая интерфейсная карта 46
сетевая операционная система 31, 52
сетевая разведка 1000
сетевая служба 51
сетевая технология 33
сетевой адаптер 46
сетевой адрес 369, 371
  выходного интерфейса 403
  следующего маршрутизатора 403
сетевой интерфейс 64, 644
сетевой монитор 400
сетевой сервер 34
сетевой уровень 113, 365
сетевой червь 1024
сетевой экран
  прикладного уровня 963
  сеансового уровня 963
  сетевого уровня 963
  с фильтрацией пакетов 963
сетевые службы 565
сеть
  LTE 721
  агрегирования трафика 129
  беспроводная 126
  виртуальная 338, 638, 646
  глобальная 30, 126
  городская 126
  дейтаграммная 127
  доступа 128
  затопление 302
  интегрируемость 148
  коммутационная 73
  компьютерная 27, 49
  корпоративная 128
  локальная 33, 126, 338
  магистральная 128
  масштабируемость 147
  мегаполиса 126
  мультисервисная 36
  на базе 127
  наложенная 129
  недогруженная 152
  оператора связи 127
  оптическая 254
  первичная 129, 241
  передачи данных 27
  персональная 813
  планирование 147
  проводная 126
  рассредоточенная 815
  расширяемость 146
  с избыточной пропускной способностью 152
  с коммутацией 127
  совместимость 148
  составная 368
  телефонная 30
  транспортная 254
  управляемость 147
  устойчивая к разрывам 880
  частная 638, 646
сеть дронов 845
сигнал
  амплитуда 187
  длина волны 188
  период 187
  спектр 188
  стартовый 47
  стоповый 47
  фаза 187
  частота 187
сигнал/шум отношение 200
сигнальный протокол 615
символьное имя 371
символьный адрес 64
симплексный канал 61

синхронизация передатчика и приемника 59, 218
синхронная цифровая иерархия SDH 250
синхронное приложение 150
система
  Т-каналов 249
  автономная 490
  доменных имен 388
  имен 386
  многотерминальная 29
  навигации 776
  обнаружения вторжений 980
  открытая 121
  пакетной обработки 28
  разделения времени 29
  управления 596, 597
  шифрования 920, 921
система доменных имен 388
сканирование портов 1000
скользящее окно 453
скорость
  передачи данных 60, 143
  пиковая 144
  предложенной нагрузки 60
  средняя 143
скремблирование 225
скрытая станция 667
слово-вызов 928, 932
слой
  менеджмента 119
  пользовательский 119
  управления 119
слот
  трибутарный 260
служба
  каталогов 51
  печати 51
  сетевая 51
  справочная 51
случайные задержки пакетов 135
случайный доступ 288
смешанная топология 64, 127
сниффер 904
совет по архитектуре интернета 123
совместимость сети 148
соглашение об уровне обслуживания 132, 597
соединение
  логическое 94
  отказ в установлении 86
  установление 84
сокет 443
сообщение 48, 99, 603
  понятие 114
  проверки непрерывности соединения 353
сообщество интернета 123
составная сеть 368
составной канал 83
  circuit 182
сота 723
  сети Wi-Fi 784
сотовые телекоммуникационные сети 720
сохранение с продвижением 91
социальный инжиниринг 902
спам 1042
спектр
  расширенный 659
спектральное разложение 188
спектральный канал 267
спецификация
  открытая 121
специфический маршрут 404, 411
список
  доступа 332, 955
список управления доступом 944
справедливое обслуживание 161
спутник
  геостационарный 775
  низкоорбитальный спутник 776
  среднеорбитальный 776

среда разделяемая 76
средневысотная 774
среднеорбитальный 776
среднесрочные характеристики сети 132
средняя скорость
  передачи данных 143
срок аренды 394
стадия
  прослушивания 329
стандарт
  комитетов и объединений 122
  международный 122
  национальный 122
  отдельных фирм 122
стандартная сетевая технология 34
стандартная топология физических связей 287
стандартный назначенный номер порта 442
станция 784
стартовый сигнал 47
статистическая оценка 137
статическая запись 302, 383
статическая маршрутизация 473
статическая страница 580
статический маршрут 411
стек
  TCP/IP 124
  коммуникационных протоколов 110
  меток 618
стек OSI
  OSI stack 123
стоповый сигнал 47
страница
  гипертекстовая 567
  динамическая 580
  статическая 580
строгая аутентификация 928
структурированная кабельная система 212
схема
  автопереговоров 313
сшивание путей 619

Т

таблица
  адресная 300, 301
  коммутации 73, 92, 97
  маршрутизации 71, 73, 402, 411
  продвижения 71, 301
  продвижения MPLS 614
  соответствия адресов 66
  фильтрации 301
тайм-аут 481
  квитанции 464
тайм-слот 238
тактовая частота 203
такт сигнала 203
тег
  виртуальной локальной сети 342
  языка разметки 568
телекоммуникационные сети 125
телефонная сеть 30
тема для обсуждения 123
теория
  автоматического управления 166
  очередей 153
терминал
  скрытый 667
терминальные мультиплексоры DWDM 271
терминальный доступ 604
тест Тьюринга
  Тьюринг 41
техника расширенного спектра 659, 708
технологии доступа 563
технология
  бесклассовой междоменной маршрутизации 378, 423
  межсетевого взаимодействия 368
  сетевая 33
тонкопленочный фильтр 272
топология
  древовидная 63, 127
  звездообразная 63, 127

кольцевая 63, 127
неполносвязная 62
полносвязная 62, 127
понятие 61
смешанная 64, 127
ячеистая 63, 252, 276
точка
встречи 502
обслуживания 353
присутствия 560
рандеву 502
доступа 784
традиционная технология NAT 970
транк 334
трансляция
протоколов 537
сетевых адресов 970
трансляция протоколов 542
трансляция сетевых адресов
базовая 971
двойная 975
и портов 971
транснациональный оператор 560
транспондер 272
транспортные услуги
безопасность 131
надежность 131
производительность 131
транспортный блок оптического канала 255
транспортный режим 1014
транспортный туннель 729
транспортный уровень 113, 365
трафик 119
инжиниринг 152, 172, 173
классификация 158
компьютерный 87
кондиционирование 152
неравномерный 99
потоковый 148
профилирование 162
пульсирующий 87
формирование 163
эластичный 149
трибутарный порт 244
трибутарный слот 260
триггерное обновление 483
триггерный кадр Wi-Fi 805
тропосферные волны 692
троянская программа 1024
туннелирование 538, 541
туннель
GTP 731
туннельный режим 1014

У

угроза
внешняя 899
неумышленная 899
понятие 898
умышленная 899
удаление метки на предпоследнем хопе 617
узкое место составного пути 60
указатель
срочности 464
улучшенная скорость передачи данных 818
умышленная угроза 899
уникальный адрес 64
унифицированный указатель ресурса 568
уплотненное волновое мультиплексирование 267
управление
безопасностью 597
доступом к среде 288
конфигурацией сети и именованием 596
логическим каналом 289
очередями 152
перегрузкой 151
управление перегрузками в сети 465
управление рисками 900
управляемость сети 147

уровень
интернета 365
канальный 112
представления 114
прикладной 364
сеансовый 114
сетевой 365
согласования 313
транспортный 113, 365
физический 112
уровень доступа к среде
MAC 289
услуги
виртуальной частной локальной сети 652
установление логического соединения 94
устройство
главное 814
компенсации дисперсии 276
подчиненное 814
физического уровня 313
устройство пользователя UE 725
учет работы сети 597
уязвимость 897

Ф

файервол 958
физический интерфейс 46
физический уровень 112
фиксированная граница адреса 372
фильтр
пользовательский 303, 332
тонкопленочный 272
фильтрация
кадра 301
понятие 954
формирование трафика 163
фотонный коммутатор 276
фрагментация 424
фрейм 367
функция
односторонняя 916

Х

хаб 297
хакер 899
характеристики
задержек пакетов 136
сети 132, 135
характеристики сети 131
хеш-функция 922
хоп 402
хорошо известный номер порта 442
хост 365
хроматическая дисперсия 211

Ц

целостность 895
центр
обмена трафиком 562
сертификации 935
централизованная справочная служба 51
централизованное сетевое приложение 56
централизованный способ назначения адреса 290
центральный офис 560
цепь 251
циклический избыточный контроль CRC 231
цифровая подпись 933
цифровой сертификат 935

Ч

частная линия Ethernet 645
частная сеть 558
частный адрес 377, 970
частота
несущая 292

частотная модуляция FM 232
частотное мультиплексирование FDM 236
частотное разделение 75
частотный план 270
червь сетевой 1024
числовой адрес 64

Ш

Шеннона формула 205
ширина спектра сигнала
  определение 190
широковещательный адрес 64, 290, 411
широковещательный шторм 302, 375
шифрование
  понятие 912
  с открытым ключом 919
  с помощью односторонней функции 916
шлюз
  безопасности 1014
  внешний 491
  понятие 365
шлюзовой протокол
  внешний 491
  внутренний 491
  пограничный 491, 492
шпионская программа 904

Э

экранированная витая пара 206
эластичный трафик 149
электронная подпись 933
элементарный канал 81, 82
энергетический бюджет линии 196
энергопотребление 679
Эрикссон Ларс Магнус 662
эстафетная передача 739
эффект Доплера 694, 777
эхо-запрос 433
эхо-ответ 433
эхо-протокол 433

Я

язык разметки гипертекста 567
ячеистая топология 63, 252, 276

Виктор Олифер, Наталья Олифер

Компьютерные сети. Принципы, технологии, протоколы

7-е издание

Руководитель дивизиона	*Ю. Сергиенко*
Ведущий редактор	*Е. Строганова*
Литературный редактор	*Ю. Широнина*
Художественный редактор	*В. Мостипан*
Корректор	*С. Беляева*
Верстка	*Л. Егорова*

Изготовлено в России. Изготовитель: ООО «Прогресс книга». Место нахождения и фактический адрес: 194044, Россия, г. Санкт-Петербург, Б. Сампсониевский пр., д. 29А, пом. 52. Тел.: +78127037373.

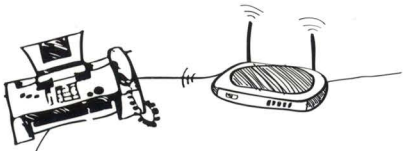
Дата изготовления: 07.2026. Наименование: книжная продукция. Срок годности: не ограничен.

Налоговая льгота — общероссийский классификатор продукции ОК 034-2014, 58.11.12.000 — Книги печатные профессиональные, технические и научные.

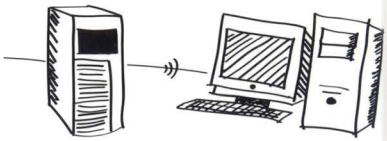
Импортер в Беларусь: ООО «ПИТЕР М», 220020, РБ, г. Минск, ул. Тимирязева, д. 121/3, к. 214, тел./факс: 208 80 01.

Подписано в печать 15.05.26. Формат 70x100/16. Бумага офсетная. Усл. п. л. 89,010. Тираж 4000. Заказ 3676.

Отпечатано в АО «Первая Образцовая типография»,
филиал «УЛЬЯНОВСКИЙ ДОМ ПЕЧАТИ». 432980, Россия, г. Ульяновск, ул. Гончарова, 14

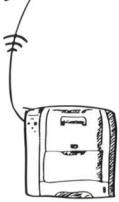


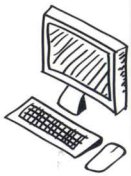
Седьмое издание



Компьютерные сети

Принципы, технологии, протоколы





Седьмое издание одного из лучших российских учебников по сетевым технологиям, переведенным на английский, португальский, испанский и китайский языки.

Секретом успеха этой книги, которая издается уже на протяжении более 25 лет, является постоянное и существенное изменение ее содержания, при котором материал каждого нового издания обновляется на 25-30 %, отражая новые тенденции и технологии быстро изменяющейся области компьютерных сетей. В то же время во всех изданиях сохраняется ядро, описывающее фундаментальные принципы и базовые механизмы современных сетевых технологий.

Коренной переработке подверглись материалы, посвященные беспроводной передаче данных, в частности глубоко и подробно рассматриваются технологии мобильных сотовых сетей и Wi-Fi. Также добавлены разделы о сетях дронов и спутниковых сетях Iridium и Starlink, описание технологий интернета вещей, таких как ZigBee и LoRaWAN.

Виктор Олифер
Наталья Олифер

Материалы книги прошли успешную проверку в сложной аудитории, состоящей из слушателей с весьма различным уровнем подготовки и кругом профессиональных интересов. Среди них — студенты и аспиранты вузов, сетевые администраторы и интеграторы, начальники отделов автоматизации и преподаватели. Учебник построен так, чтобы начинающий получил основу для дальнейшего изучения, а специалист смог систематизировать и актуализировать имеющиеся знания. Чтобы не делать книгу слишком большой, а также из-за снижения актуальности тех или иных технологий часть материалов предыдущих изданий перенесена на сайт www.olifer.co.uk.

Рекомендовано Министерством образования и науки Российской Федерации в качестве учебного пособия для студентов высших учебных заведений, обучающихся по направлению «Информатика и вычислительная техника» и по специальностям «Вычислительные машины, комплексы, системы и сети», «Автоматизированные машины, комплексы, системы и сети», «Программное обеспечение вычислительной техники и автоматизированных систем».

ПИТЕР®

WWW.PITER.COM
интернет-магазин

Заказ книг:
(812) 703-73-74
books@piter.com

PiterBooks
PiterForPeople
ThePiterBooks
Company/piter

ISBN:978-5-4461-4566-9

