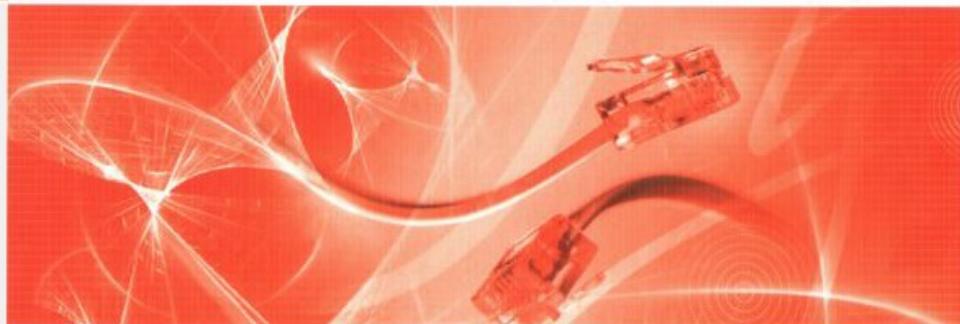




В. Олифер Н. Олифер



Основы компьютерных сетей

- Основы построения сетей ■
- Технологии локальных и глобальных сетей ■
- Обзор популярных сетевых служб и сервисов ■



В. Олифер, Н. Олифер

Основы компьютерных сетей



Москва · Санкт-Петербург · Нижний Новгород · Воронеж
Ростов-на-Дону · Екатеринбург · Самара · Новосибирск
Киев · Харьков · Минск
2009

*Виктор Григорьевич Олифер
Наталья Алексеевна Олифер*

Основы компьютерных сетей

Серия «Учебное пособие»

Заведующий редакцией
Руководитель проекта
Ведущий редактор
Литературный редактор
Художественный редактор
Корректор
Верстка

*A. Сандрыкин
A. Юрченко
O. Некруткина
A. Жданов
L. Адуевская
И. Тимофеева
O. Шакиров*

**ББК 32.202я7
УДК 004.7(075)**

Олифер В. Г., Олифер Н. А.
О-54 Основы компьютерных сетей. — СПб.: Питер, 2009. — 352 с.: ил.

ISBN 978-5-49807-218-0

Издание представляет собой краткий учебный курс, в котором последовательно рассматриваются основные аспекты архитектуры и технологии современных компьютерных сетей. В книге освещены вопросы главных концепций, являющихся фундаментом компьютерных сетей, технологии проводных, беспроводных локальных сетей, составной сети (Интернета), Глобальной сети, популярных сетевых услуг и сервисов.

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.

ISBN 978-5-49807-218-0

© ООО «Лидер», 2009

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.

ООО «Лидер», 194044, Санкт-Петербург, Б. Сампсониевский пр., д. 29а.
Налоговая льгота — общероссийский классификатор продукции ОК 005-93,
том 2; 95 3005 — литература учебная.

Подписано в печать 14.04.09. Формат 60x88/16. Усл. п. л. 25. Тираж 5000. Заказ 15308.
Отпечатано по технологии СоТ в ОАО «Печатный двор» им. А. М. Горького.
197110, Санкт-Петербург, Чкаловский пр., д. 15.

Краткое содержание

Предисловие	13
Глава 1. Основы построения сетей	17
Глава 2. Технологии локальных сетей.	111
Глава 3. Сети TCP/IP	173
Глава 4. Технологии глобальных сетей	263
Глава 5. Сетевые услуги	309

Содержание

ГЛАВА 1. Основы построения сетей	17
Сеть с высоты птичьего полета.....	18
Веб-серфинг как первый опыт	18
Интернет. Вид сверху.....	19
Путешествие пакетов по Сети.....	21
Простейший случай связи двух компьютеров	24
Совместное использование ресурсов	24
Связь компьютера с периферийным устройством	25
Доступ к периферийному устройству через сеть	28
Передача данных по линиям связи	31
Сетевое программное обеспечение	34
Сетевые службы и сервисы	34
Сетевая операционная система	36
Сетевые приложения.....	39
Проблемы связи нескольких компьютеров	42
Топология физических связей	42
Адресация узлов сети.....	45
Коммутация.....	49
Определение потоков данных.....	50
Определение маршрутов.....	52
Продвижение данных	54
Мультиплексирование и демультиплексирование	57
Разделение физической среды	59
Коммутация пакетов и каналов.....	62
Сети с коммутацией каналов	63
Элементарный канал	63
Составной канал.....	65

Неэффективность при передаче пульсирующего трафика	70
Сети с коммутацией пакетов	70
Буферизация пакетов	73
Дейтаграммная передача	74
Передача с установлением логического соединения	76
Передача с установлением виртуального канала	77
Сравнение сетей с коммутацией пакетов и каналов	80
Типы компьютерных сетей.....	82
Глобальные сети	82
Локальные сети	85
Составные сети	87
Телекоммуникационные сети.....	90
Сети операторов связи.....	92
Корпоративные сети.....	94
Стандартизация сетей	97
Многоуровневый подход	97
Модель OSI.....	101
Функции уровней модели OSI	105
Распределение функций между различными элементами сети ...	108
Стандартные стеки протоколов	109
ГЛАВА 2. Технологии локальных сетей.....	111
Особенности локальных сетей	112
Локальные сети на разделяемой среде	113
Ethernet на коаксиальном кабеле	114
Ethernet на витой паре	123
Сети Token Ring и FDDI	125
Достоинства и недостатки разделяемой среды	128
Коммутируемые сети Ethernet	130
Принцип работы коммутаторов Ethernet	130
Протокол покрывающего дерева	135
Скоростные версии Ethernet.....	138
Кабели и методы кодирования	139
Классический вариант Ethernet.....	148

Fast Ethernet	149
Gigabit Ethernet	153
10G Ethernet	156
Виртуальные локальные сети	157
Пользовательские фильтры	157
Логическое разделение сети на виртуальные локальные сети	159
Беспроводные локальные сети.....	163
Стандарты IEEE 802.11	165
Персональные сети Bluetooth	169
ГЛАВА 3. Сети TCP/IP.....	173
Стек протоколов TCP/IP.....	173
Адресация в сетях TCP/IP	177
Типы адресов стека TCP/IP	177
Формат IP-адреса	179
Классы IP-адресов	180
Использование масок при IP-адресации	183
Порядок назначения IP-адресов и технология CIDR	184
Протокол ARP	187
Доменные имена	190
Система DNS.....	192
Протокол DHCP	194
Протокол межсетевого взаимодействия	197
Формат IP-пакета	197
Таблица маршрутизации	199
Маршрутизация без масок	203
Пример взаимодействия протоколов IP, ARP, Ethernet и DNS	205
Структуризация сетей на основе масок	211
Маршрутизация с масками.....	213
CIDR и маршрутизация	216
Фрагментация IP-пакетов.....	218
Протоколы транспортного уровня TCP и UDP.....	222
Порты и сокеты.....	223
Протокол UDP	226

Протокол TCP и TCP-сегменты.....	227
Логические соединения — основа надежности TCP	229
Повторная передача и скользящее окно	233
Реализация метода скользящего окна в протоколе TCP	235
Управление потоком.....	239
Протоколы маршрутизации	242
Общие свойства протоколов маршрутизации.....	242
Протокол OSPF	245
Взаимодействие протоколов маршрутизации	248
Внешние и внутренние шлюзовые протоколы.....	250
Вспомогательные протоколы и средства стека TCP/IP	252
Протокол ICMP	252
Утилита traceroute.....	254
Утилита ping	256
Протокол NAT	257
ГЛАВА 4. Технологии глобальных сетей	263
Первичные сети	263
Сети PDH.....	265
Сети SONET/SDH.....	268
Сети DWDM	271
Сети OTN.....	273
Технология Frame Relay	273
История стандарта	273
Техника продвижения кадров	274
Гарантии пропускной способности	277
Технология ATM	279
Ячейки ATM	280
Оцифровывание голоса.....	280
Виртуальные каналы ATM	284
Категории услуг ATM	284
Технология MPLS.....	285
LSR и таблица продвижения данных	286
Пути коммутации по меткам	288

Заголовок MPLS и технологии канального уровня.....	290
Отказоустойчивость MPLS.....	291
Области применения технологии MPLS	292
Глобальные сети IP	294
Структура глобальной сети IP	294
Протоколы HDLC и PPP	296
Carrier Ethernet — Ethernet операторского класса	298
Движущие силы Carrier Ethernet.....	298
Ethernet на основе MPLS.....	299
Ethernet на основе Ethernet, или Carrier Ethernet Transport	301
Удаленный доступ	304
Проблемы удаленного доступа.....	304
Схемы удаленного доступа.....	305
ГЛАВА 5. Сетевые услуги	309
Электронная почта	309
Электронные сообщения.....	310
Протокол SMTP	310
Непосредственное взаимодействие клиента и сервера.....	311
Схема с выделенным почтовым сервером	312
Схема с двумя почтовыми серверами-посредниками.....	315
Протоколы POP3 и IMAP	316
Веб-служба	317
Веб- и HTML-страницы	318
URL.....	319
Веб-клиент и веб-сервер.....	319
Протокол HTTP	321
Формат HTTP-сообщений.....	322
Динамические веб-страницы.....	324
Протокол передачи файлов	325
Основные модули службы FTP	325
Управляющий сеанс и сеанс передачи данных	326
Команды для взаимодействия FTP-клиента с FTP-сервером	327

Системы управления сетью и протокол SNMP	327
Схема «менеджер — агент — управляемый объект».....	328
Протокол SNMP	329
Структура систем управления	329
Протокол telnet.....	331
Службы сетевой безопасности	332
Безопасность компьютера и сетевая безопасность.....	332
Конфиденциальность, целостность и доступность данных.....	333
Угрозы, атаки, риски.....	335
Шифрование, сертификат, электронная подпись.....	336
Идентификация, аутентификация, авторизация, аудит	343
Технология защищенного канала	346
Политика безопасности.....	347
Рекомендуемая литература	349

« »

,

,

,

,

,

,

,

,

,

,

« »

,

,

,

,

2009

comScore

ARPANET (

),

,

(

)

«

»

,

,

,

,

,

,

«Компьютерных сетей», многое изменилось. Главное — мир сетевых технологий стал существенно более однородным. К технологии TCP/IP, которая уже сравнительно давно доминирует в своем секторе, присоединилась технология локальных сетей Ethernet, одержавшая безоговорочную победу над конкурировавшими когда-то с ней Token Ring и FDDI. Ушли в прошлое концентраторы, мосты и все другое, что связано с разделением среды в проводных сетях. Вместе с ними из нашей книги исчезло множество страниц, на которых мы обсуждали особенности этих технологий и оборудования. Подобные же изменения произошли и в глобальных сетях. Кто бы мог подумать еще несколько лет назад, что провайдеры телекоммуникационных сетей станут рассматривать Ethernet (более точно Carrier Ethernet) в качестве основы для построения своих глобальных сетей! Но это свершившийся факт, и мы как авторы не можем его игнорировать.

Для кого эта книга

Среди потенциальных читателей этой книги авторы, прежде всего, видят:

- студентов, 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» в качестве начального курса;
- студентов, преподавателей и аспирантов смежных специальностей, для которых курс «Компьютерные сети» не является профилирующим;
- студентов-заочников или обучающихся по системе дистанционного образования;
- специалистов, чья профессиональная деятельность связана с сетями лишь косвенно, например, административных руководителей предприятий, прикладных программистов, менеджеров по продажам сетевого оборудования, которые нуждаются в базовых знаниях о компьютерных сетях.

А еще мы хотели бы видеть среди своих читателей тех пытливых пользователей, которые, уходя с работы, выключают свой компьютер, а придя домой, снова выходят в Сеть, чтобы поговорить по Skype с другом, купить билет на самолет, написать письма, узнать о погоде на завтра, проверить, не появилась ли новая интересная вакансия, посмотреть теннисный матч, который не показывают по телевизору, — да мало ли чем еще можно заняться в Интернете. Конечно, для работы в Сети вовсе не обязательно знать, как там все внутри устроено. Но иногда даже самые далекие от техники люди вдруг чувствуют необходимость прояснить для себя загадку: «А что же все-таки это такое — IP-адрес?» Такой читатель, мы надеемся, сможет в этой книге найти и, главное, понять ответы на многие свои вопросы.

Структура книги

Книга состоит из пяти глав.

Глава 1 «Основы построения сетей» посвящена изложению главных концепций, являющихся фундаментом компьютерных сетей. Очень важно, чтобы читатель хорошо их понял, прежде чем перейдет к изучению конкретных технологий.

В главе 2 «Технологии локальных сетей» рассматриваются технологии как проводных, так и беспроводных локальных сетей. Основное внимание уделяется коммутируемой версии Ethernet, приводится также краткое описание других технологий на разделяемой среде, представляющих сегодня историко-научный интерес.

Глава 3 «Сети TCP/IP» представляет собой компактное введение в технологию составной сети, или интернета (с маленькой буквы). Здесь приведена общая характеристика стека TCP/IP, способы адресации, продвижения и маршрутизации пакетов в сетях IP.

В главе 4 «Технологии глобальных сетей» описываются технологии Frame Relay, ATM, MPLS и Carrier Ethernet, специально созданные для построения сетей, покрывающих большие территории и объединяющих большое количество узлов. В этой главе читатель также познакомится с первичными сетями, составляющими фундамент любой сети.

Глава 5 «Сетевые услуги» посвящена самым популярным сетевым службам и сервисам, таким как электронная почта, веб-служба (WWW), службы передачи файлов (FTP), удаленного управления (telnet) и управления сетью (SNMP). Здесь рассмотрены также основные принципы организации службы безопасности сети.

Веб-сайт

На сайте www.olifer.co.uk вы можете найти дополнительную информацию, связанную с этой и другими нашими книгами. Там размещены все иллюстрации из книг, презентации в форматах PowerPoint, методические рекомендации для преподавателей, ответы на вопросы к главам книги, дополнительные вопросы и примеры, которые могут быть использованы в качестве тем курсовых работ, и другие ресурсы. На сайте также публикуются мнения, замечания и вопросы читателей, замеченные опечатки и ошибки.

Мы с благодарностью примем ваши отзывы по адресам victor@olifer.co.uk и natalia@olifer.co.uk.

От издательства

Подробную информацию о наших книгах вы найдете на веб-сайте издательства <http://www.piter.com>. Там же вы можете оставить ваши отзывы и пожелания.

Мы будем рады узнать ваше мнение!

ГЛАВА 1. Основы построения сетей

Господа, начните, пожалуйста, с самой сути! Умоляю вас, не надо деталей!

Артур Конан Дойль

Меняйте ваши мнения, сохраняйте принципы: меняйте листья, сохраняйте корни.

Виктор Гюго

Понимание принципов важно при изучении любого явления или объекта, оно помогает осознать и более эффективно использовать отдельные факты и детали, связать их в стройную систему. Именно об этом говорит известное выражение «Знание нескольких принципов освобождает от запоминания множества фактов».

Когда вы приступаете к изучению конкретных сетевых технологий, таких как Ethernet, IP или ATM, то очень скоро начинаете понимать, что у этих технологий много общего. Это общее состоит в том, что разные технологии призваны решать сходные задачи. К примеру, практически в любой сетевой технологии решаются среди прочих такие задачи, как выбор эффективной системы адресации, способной компактно и в то же время однозначно идентифицировать многочисленные узлы и пользователей сети, нахождение маршрута между конечными узлами сети с произвольной структурой и т. п.

Общие задачи естественным образом породили и общие подходы к их решению. Так, например, в основе многих систем адресации лежит фундаментальный принцип иерархической организации адресов. А для нахождения рационального маршрута во многих технологиях используется принцип кратчайшего маршрута.

Понятно, что схожесть этих и других, общих для разных технологий задач не делает их тождественными. Они имеют специфические особенности, присущие каждой технологии, так что нельзя механически перенести знания из одной технологии в другую. Например, как бы глубоко вы ни разбирались в адресации, маршрутизации и мультиплексировании в IP-сетях, это не позволит вам в деталях судить о том, как решаются эти задачи в сетях ATM. Однако знание базовых принципов, на которых построены обе эти технологии, сделает процесс их изучения более легким и эффективным.

Сеть с высоты птичьего полета

Веб-серфинг как первый опыт

Скорее всего, наши читатели впервые познакомились с компьютерными сетями, получив доступ в Интернет для поиска и просмотра информации. Именно поэтому мы и начнем изучение сетей, опираясь на этот, возможно, минимальный опыт наших читателей.

Для доступа к информационным ресурсам Интернета необходимо, чтобы ваш компьютер был подключен к Сети¹. Мы сейчас не будем подробно останавливаться на том, что стоит за словом «подключен», отметим лишь, что это означает, что на компьютере установлены все программные и аппаратные средства, необходимые для работы в Сети, и существует физическое соединение (проводное или беспроводное) этого компьютера с одним из провайдеров Интернета, то есть компанией, компьютерная сеть которой является частью Интернета.



Рис. 1.1. Главное окно браузера Chrome компании Google

Для поиска и просмотра информации в Интернете на компьютере должна быть установлена программа под названием **веб-браузер** (web browser). Существуют различные браузеры, и, с точки зрения пользователя, все они работают примерно одинаково. На рис. 1.1 показано окно, которое выводит на экран новый браузер Chrome компании Google, выпущенный в сентябре 2008

¹ Вместо «сеть Интернет» часто говорят и пишут просто «Сеть» или «Интернет».

года. В верхней части экрана расположено поле адреса, то есть поле, в котором вы набираете имя интересующего вас **веб-сайта** (web site) — источника информации, находящегося в Интернете. Пусть в данном случае вы обратились к сайту www.olifer.co.uk — информационному ресурсу авторов этой книги. После того как вы набрали имя и нажали клавишу **Enter**, ваш компьютер отправляет в сеть **запрос** к указанному вами источнику информации. Запрос путешествует по Сети, пока не достигнет компьютера, на котором расположен веб-сайт www.olifer.co.uk. Этот запрос, как и все остальные запросы, поступающие на данный веб-сайт, принимается и обслуживается специальной программой, называемой **веб-сервером** (web server). По отношению к веб-серверу браузеры выступают **клиентами**. В ответ на поступивший запрос веб-сервер сайта www.olifer.co.uk передает информацию, размещенную на его главной странице, вашему браузеру, который и выводит ее на экран вашего компьютера в том виде, как показано на рисунке.

Интернет. Вид сверху

Давайте теперь попробуем взглянуть «сверху» на то, что же происходило в вашем компьютере и в Интернете в те несколько секунд (или, если вам не повезло, минут), в течение которых обрабатывался ваш запрос. Наш «птичий полет» будет проходить над картой Интернета, но представить такую карту на бумаге сегодня весьма непросто. Дело в том, что современный Интернет — это сеть, связывающая между собою сети, расположенные по всему миру (поэтому Интернет называют также сетью сетей). Двадцать лет назад, когда число сетей, объединенных в Интернет, исчислялось сотнями, такую карту еще можно было представить в виде рисунка на одной странице (рис. 1.2).

На этом рисунке каждый овал представляет собой одну сеть, кружок — **маршрутизатор**, то есть коммуникационное устройство, которое передает информацию между сетями, а линии соответствуют физическим каналам связи, таким как, например, кабели. Каждая из сетей, представленных на рис. 1.2, также имеет свою структуру, которая во многом повторяет структуру Интернета, так как в ней тоже есть линии связи и коммуникационные устройства, только вместо сетей они соединяют **конечные узлы**. В качестве конечных узлов могут выступать самые разнообразные вычислительные устройства, различающиеся функциональной специализацией (телефоны, компьютеры, встроенные микропроцессорные устройства), а также размерами и вычислительной мощностью (ноутбуки, персональные компьютеры, мэйнфреймы).

В зависимости от того, какую роль играют в сети конечные узлы, различают **серверы**¹, на которых хранится информация, и **клиентские узлы**, с помощью которых пользователи получают доступ к информации, хранящейся на серверах.

¹ Ранее было сказано, что сервер — это программа, принимающая и обслуживающая запросы от других программ — клиентов. Этими же терминами обозначают компьютеры, на которых выполняются серверные и клиентские программы.

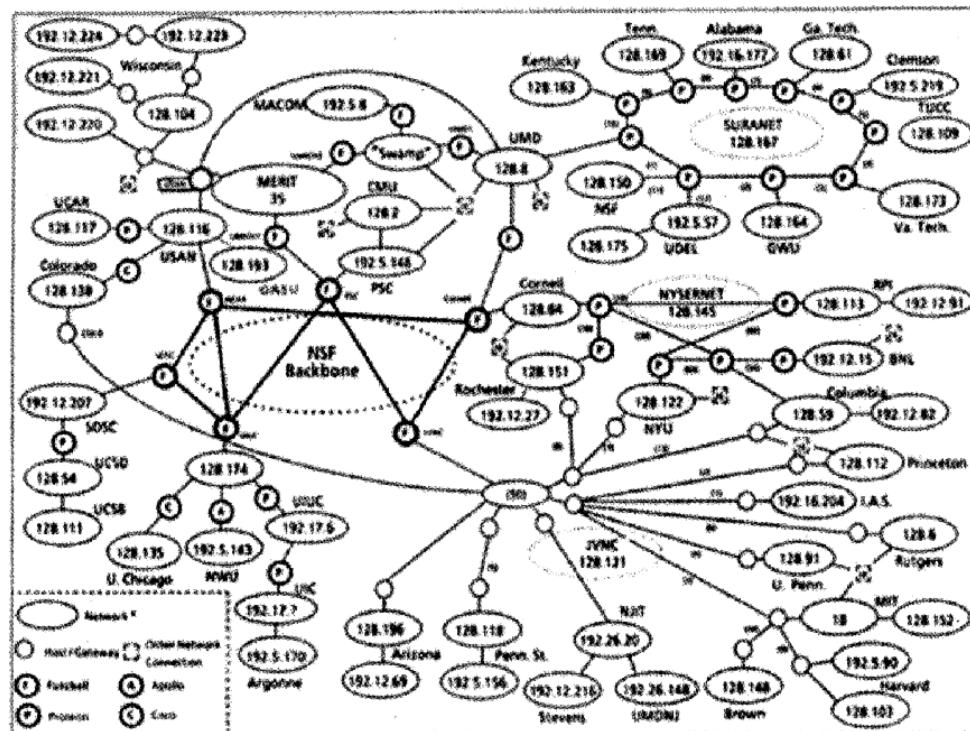


Рис. 1.2. Структура Интернета в 1987 году

Из рисунка хорошо видно, что в 1987 году Интернет не имел какой-либо регулярной структуры, сети были связаны между собой достаточно хаотичным образом и между многими из них существовало несколько возможных маршрутов обмена данными. Такая децентрализованная структура Интернета была выбрана не случайно, это был один из архитектурных принципов его построения, и этот принцип хорошо зарекомендовал себя в процессе бурного роста Интернета, поэтому и сегодня структура Интернета остается такой же хаотичной, какой была на заре его развития.

Однако сейчас визуально представить Интернет стало нетривиальной задачей, потому что он превратился во **Всемирную паутину**¹, связывающую миллионы сетей. Над визуализацией Интернета работают ученые и энтузиасты, и один из вариантов такого представления, полученный в результате анализа компьютерной программой трафика между узлами сети, показан на рис. 1.3.

¹ Точнее, Всемирная паутина (World Wide Web, WWW) — это название одного из самых популярных сервисов Интернета, который собственно и превратил Интернет в элемент современной цивилизации.

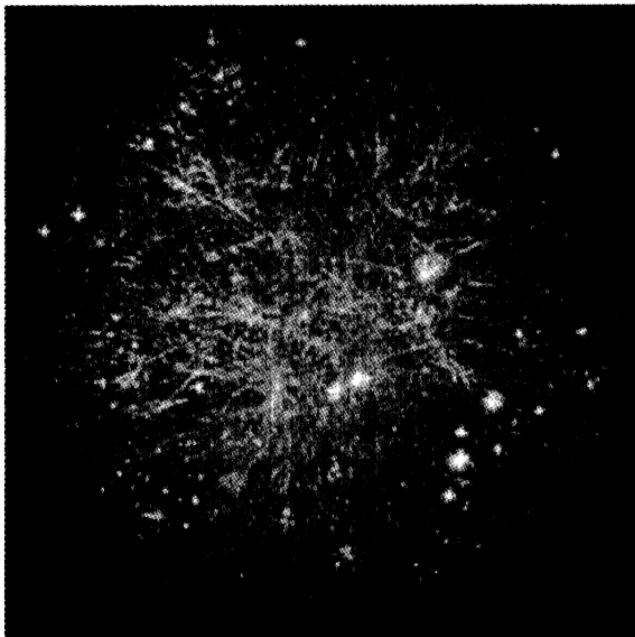


Рис. 1.3. Интернет сегодня

Полученное таким образом изображение, наверное, более всего похоже на картину звездного неба, где звездами являются сети.

Другой аналогией, помогающей представить сегодняшний Интернет в целом, могла бы стать всемирная карта дорог, включающая дороги всех типов: магистральные, проселочные, городские улицы и т. п. Перекрестки дорог в этой аналогии соответствуют коммуникационным устройствам, которые направляют машины (данные) на ту или иную дорогу.

Вы можете пользоваться любой из этих аналогий, главное — чтобы она помогла вам представить современный Интернет как очень большое количество компьютеров, связанных между собой сложной сетью каналов связи с нерегулярной и избыточной структурой через промежуточные узлы — маршрутизаторы. И где-то к этой гигантской паутине подключены клиентский и серверный компьютеры из нашего примера. При этом они могут располагаться как на разных континентах, так и в одном и том же здании, принцип обмена информацией от этого не изменится, изменится только количество пройденных маршрутизаторов и каналов связи.

Путешествие пакетов по Сети

Итак, вы ввели адрес <http://www.olifer.co.uk> и нажали клавишу Enter. Представим, что мы наделены способностью видеть, как данные перемещаются по сети, а также отличать «свои» данные (то есть те, которые связаны с нашей

операций) от всех остальных. В первую очередь мы заметим, что данные передаются по сети не в виде сплошного потока битов, а порциями — **пакетами**. Каждый пакет начинается с **заголовка** — набора служебной информации, такой, например, как адреса отправителя и получателя.

Некоторые компьютеры становятся активными периодически, направляя в сеть один или несколько пакетов и после этого замирая на довольно длительное время, — это пользовательские компьютеры, которые активизируются синхронно с действиями пользователя. Другие компьютеры постоянно активны, выбрасывая в сеть тысячи пакетов в секунду, — это серверы, которые обслуживают запросы пользователей, снабжая их информацией. В целом передвижение пакетов по сети очень напоминает автомобильное движение — некоторые каналы и маршрутизаторы загружены почти до предела или даже перегружены, в то время как другие свободны и простирают.

Но вот появились и первые «наши» пакеты, пусть они будут для нас белыми и черными (рис. 1.4). Белые — это те, которые идут от нашего клиентского компьютера, а черные — те, которые ему адресованы. Но вот какая странность: первый белый пакет направился отнюдь не в сторону сервера www.olifer.co.uk, как можно было бы предположить, а быстро «свернулся» к серверу, который находится в сети нашего провайдера. То же произошло с первым черным пакетом — он пришел в наш компьютер не от сервера www.olifer.co.uk, а от того же сервера провайдера.

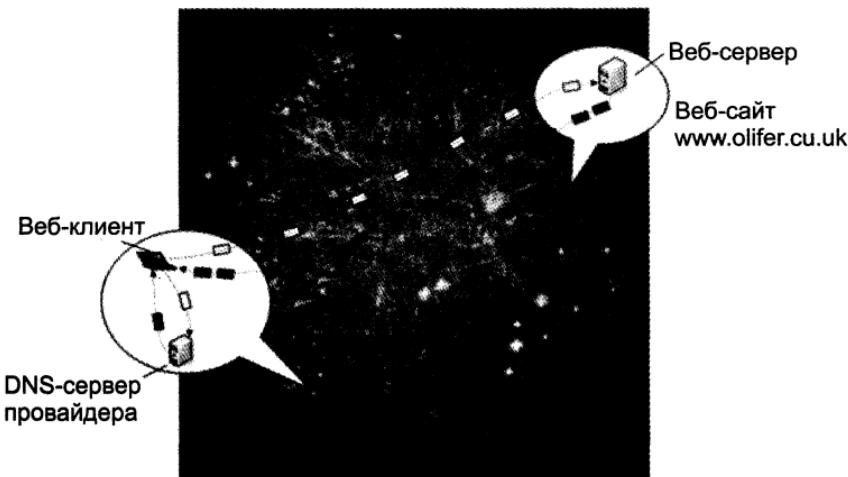


Рис. 1.4. Прохождение пакетов через Интернет

И только после такого предварительного обмена пакетами мы наконец увидели цепочку белых пакетов, идущих от нашего компьютера к серверу www.olifer.co.uk, а также более многочисленные пульсирующие колонны черных пакетов, идущих от сервера в обратном направлении. Присмотревшись к содержанию черных пакетов, мы увидим, что это кусочки нескольких картинок,

из которых состоит первая страница сайта www.olifer.co.uk. По мере поступления черных пакетов на экране нашего компьютера из этих кусочков складывается полное изображение страницы.

Однако что же все-таки происходило в первые мгновения после нажатия клавиши **Enter**? Может быть, сеть дала сбой, и первый пакет просто заблудился? Или же это нормальная ситуация? Отгадка кроется в тех белых пакетах, которые потом пошли к нужному серверу. Если мы посмотрим на них внимательнее, то заметим, что заголовки всех этих пакетов содержат один и тот же адрес назначения — 69.73.175.185. Адреса такого вида называются **IP-адресами**. Именно IP-адрес 69.73.175.185, а не имя www.olifer.co.uk использовали маршрутизаторы Интернета, чтобы направить белые пакеты в правильном направлении. Прежде чем отправить пакеты в Сеть, браузер должен был получить ответ на вопрос: «Какой IP-адрес соответствует имени www.olifer.co.uk?» Как раз этот вопрос и содержался в самом первом белом пакете, отправленном браузером по заранее известному ему адресу сервера провайдера, который хранит таблицу соответствия имен и IP-адресов (так называемый **DNS-сервер**). А первый черный пакет доставил браузеру ответ DNS-сервера в форме «Это адрес 69.73.175.185».

Разрешив первую загадку, посмотрим теперь на белые и черные пакеты, которые передавались через Интернет между клиентским и серверным компьютерами. Первое, что можно о них сказать, — каждый белый пакет порождал один или несколько черных пакетов в ответ. То есть между клиентом и сервером происходил **диалог** — клиент что-то запрашивал у сервера, а сервер отвечал. Такие диалоги в сети происходят по определенным правилам, которые называются **протоколом**. Диалог между браузером (веб-клиентом) и веб-сервером всегда происходит по специально разработанному для этой цели стандартному протоколу **HTTP** (Hypertext Transfer Protocol — протокол передачи гипертекста).

Первый белый пакет, достигший сервера www.olifer.co.uk, сообщил ему, какими возможностями по отображению информации располагает клиент, в частности, какого формата картинки он может воспроизводить на экране и поддерживает ли он Flash-анимацию. В пакете также содержался запрос к серверу относительно структуры данных, размещенных на сайте. Сервер ответил (посыпая черные пакеты), что первая страница сайта www.olifer.co.uk состоит из нескольких элементов, в том числе трех картинок: `packet.gif`, `Title_rus.gif` и `Title_eng.gif`. Кроме того, веб-сервер передал содержание текстовых элементов страницы, а также сведения о себе, например, о том, какое программное обеспечение поддерживает этот сайт и какие режимы передачи информации он использует.

После этого браузер нашего клиентского компьютера по очереди запросил передачу каждой из трех картинок. Заметим, что передача каждой картинки потребовала пересылки достаточно большого числа черных пакетов, при этом чем больше картинка, тем больше пакетов нужно, чтобы ее передать.

Это легко объяснить, если знать, что в компьютерных сетях размер пакета ограничен, как правило, значением в 1500 байт. Использование пакетов сравнительно небольших размеров — один из краеугольных принципов работы пакетных сетей. За счет этого сеть может более эффективно использовать каналы связи, так как о ни не монополизируются надолго одной парой абонентов (на все время передачи «длинного» пакета), а благодаря поочередной передачи «коротких» пакетов разделяются между многими абонентами.

Современные скорости передачи битов данных по каналам связи очень высоки, например, сегодня 10 Гбит/с (гигабит равен тысяче миллионов битов) — это обычная скорость магистральных каналов Интернета. Пакеты путешествуют по сети очень быстро, среднее время оборота пакета (то есть его передачи от клиента серверу и обратно) обычно составляет от нескольких десятков до нескольких сотен миллисекунд. Поэтому страница веб-сайта на экране вашего браузера возникает чаще всего без особых задержек, несмотря на большое количество пакетов, требующихся для ее передачи через сеть. Например, первая страница сайта www.olifer.co.uk загрузилась в пробном teste, который мы провели во время написания этих строк, за полторы секунды, а на ее передачу было затрачено около 300 пакетов. Географическое расстояние между точками, где были расположены клиентский и серверный компьютеры в этом эксперименте, составило около 8 тысяч километров. Но сказать наверняка, сколько километров проходили пакеты по каналам связи, непросто, так как выбор маршрута в Интернете выполняется каждым маршрутизатором независимо от других и критерии такого выбора часто зависят от решения администратора сети каждого отдельного провайдера. Проще выяснить, сколько промежуточных маршрутизаторов было проидено — в данном случае их было 15.

На этом мы закончим наше «поверхностное» наблюдение за работой Интернета. Наверняка эта первая попытка описать работу компьютерной сети породила у читателя множество вопросов, и это естественно. Странно, если бы было по-другому: ведь мы только начинаем изучение компьютерных сетей, и все еще впереди. После того как мы рассмотрели компьютерную сеть «с высоты птичьего полета» давайте попробуем «нырнуть на дно» сети и посмотреть более детально, что же происходит там, «в глубине», на уровне элементарного соединения двух компьютеров.

Простейший случай связи 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).

Понятно, что для достижения этой цели аппаратное и программное обеспечение автономно работающих компьютеров должно быть дополнено некоторыми *специальными сетевыми средствами*. Давайте выясним, что они могут собой представлять, на простейшем примере, когда сеть образована только двумя компьютерами, а разделяемым ресурсом является принтер (рис. 1.5). Но еще раньше мы рассмотрим, как взаимодействуют друг с другом компьютер и локально подключенное к нему периферийное устройство (ПУ).

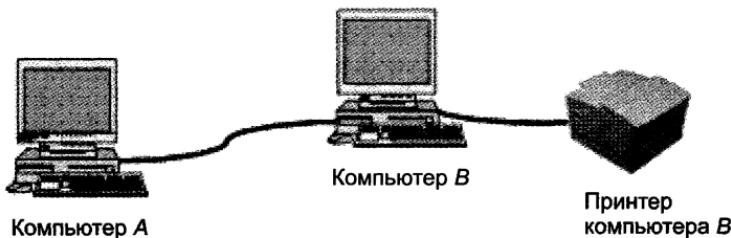


Рис. 1.5. Совместное использование принтера

Связь компьютера с периферийным устройством

Для организации связи между компьютером и периферийным устройством в обоих этих устройствах предусмотрены внешние¹ интерфейсы (рис. 1.6).

Интерфейс — в широком смысле — формально определенная логическая и (или) физическая граница между взаимодействующими независимыми объектами. Интерфейс задает параметры, процедуры и характеристики взаимодействия объектов.

Разделяют физический и логический интерфейсы.

¹ Наряду с внешними электронные устройства могут использовать внутренние интерфейсы, определяющие логические и физические границы между входящими в их состав модулями. Так, например, известный интерфейс «общая шина» является внутренним интерфейсом компьютера, связывающим оперативную память, процессор и другие блоки компьютера.

- **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.
- **Логический интерфейс** (называемый также **протоколом**) — это набор информационных сообщений определенного формата, которыми обмениваются два устройства или две программы (в данном случае компьютер и периферийное устройство), а также набор правил, определяющих логику обмена этими сообщениями.

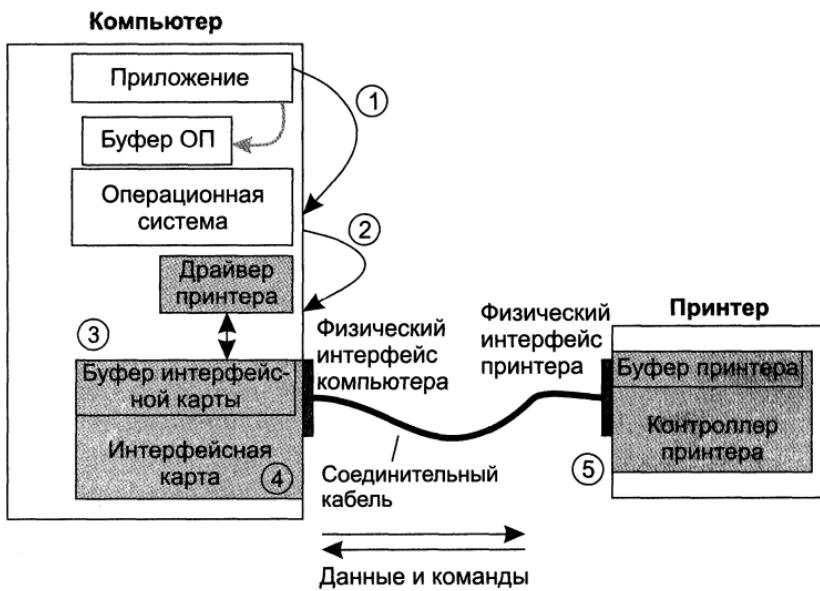


Рис. 1.6. Связь компьютера с периферийным устройством

В компьютере функции внешнего интерфейса реализуется совокупностью аппаратных и программных средств: **интерфейсной картой** (аппаратное устройство) и специальной программой, управляющей этой картой, которую называют **драйвером**. Как правило, драйвер не может быть запущен на выполнение непосредственно приложением, это может быть сделано только **операционной системой** в ответ на запрос, поступивший от приложения.

В ПУ интерфейс чаще всего полностью реализуется аппаратным устройством — **контроллером ПУ**, хотя встречаются и программно-управляемые контроллеры, предназначенные для управления современными принтерами, обладающими более сложной логикой. Контроллер периферийного устройства принимает 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.

Для того чтобы устройство можно было подключить к компьютеру, оно должно обладать интерфейсом, совместимым с интерфейсом компьютера. Как и во всех других случаях, совместимость проще всего обеспечить путем стандартизации. Примерами стандартных интерфейсов, используемых в компьютерах, являются интерфейс Centronics, предназначенный, как правило, для подключения принтеров, а также интерфейсы RS-232C (еще известный как COM-порт) и USB. Интерфейс Centronics является параллельным, данные в нем передаются байтами. Интерфейсы RS-232C и USB являются последовательными, данные в них передаются битами. Эти два интерфейса имеют более универсальное назначение — они поддерживаются не только принтерами, но и графопостроителями, манипуляторами типа «мышь» и многими другими устройствами.

Как мы видим, одна и та же интерфейсная карта, реализующая один из стандартных интерфейсов, может быть использована для связи с периферийными устройствами разного типа, поддерживающими данный интерфейс. Однако при этом для каждого типа этих устройств должен быть разработан собственный драйвер. Более того, даже для одной и той же модели ПУ может существовать несколько различных драйверов, каждый из которых реализует собственный алгоритм управления данным устройством. Например, подключая к компьютеру мобильный телефон в качестве периферийного устройства через интерфейс USB, мы можем использовать два различных драйвера. Один драйвер управляет телефоном как внешним запоминающим устройством, позволяя обмениваться с ним файлами, например, фотографиями или музыкой в формате MP3, в то время как другой драйвер предназначен для конфигурирования телефона с помощью экрана и клавиатуры компьютера, которые гораздо удобнее и функциональнее, чем экран и клавиатура телефона.

А сейчас давайте вернемся к компьютеру, к которому подключен принтер, и посмотрим, как взаимодействуют интерфейсная карта, драйвер и контроллер принтера в то время, когда приложение выводит данные на печать (см. рис. 1.6).

1. Приложение обращается с запросом на выполнение операции ввода-вывода к операционной системе. В запросе указываются адрес данных в оперативной памяти (адрес буфера ОП) и идентифицирующая информация о требуемых периферийном устройстве и операции.
2. Получив запрос, операционная система запускает драйвер принтера. С этого момента все дальнейшие действия по выполнению операции ввода-вывода со стороны компьютера реализуются только драйвером принтера и работающей под его управлением интерфейсной картой 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, которая побайтно передает их по сети контроллеру периферийного устройства.
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 синхронизируют 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.

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.

Доступ к периферийному устройству через сеть

Вернемся к ситуации, представленной на рис. 1.5: пользователю, работающему с некоторым приложением на компьютере *A*, требуется распечатать текст, но принтер подключен к компьютеру *B*. Мы будем считать, что управление принтером со стороны компьютера *B* осуществляется по только что описанной схеме. 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*?

Очевидно, что для этого как минимум необходимо оснастить оба компьютера средствами **межмашинной связи** (MMC), то есть средствами, которые

позволяют им обмениваться информацией. Межмашинная связь может быть построена на тех же принципах, что и связь компьютера с периферийным устройством. Это означает, что оба компьютера должны иметь совместимые внешние интерфейсы, для чего в обоих компьютерах устанавливаются соответствующие интерфейсные карты, связанные кабелем, а также драйверы, управляющие работой интерфейсных карт. В таких случаях говорят о создании линии, или **канала, связи** между двумя компьютерами.

Пусть по аналогии с рассмотренным ранее примером связи компьютер—принтер связь компьютер—компьютер будет также осуществляться через последовательный интерфейс (такой, например, как СОМ-порт). То есть в каждом из компьютеров может быть установлена интерфейсная карта (ИК) подобная интерфейсной карте, использованной для связи компьютера с принтером. Однако теперь вместо драйвера принтера эти интерфейсные карты управляются другими драйверами, решающими другую задачу. В простейшем случае функции драйверов межмашинной связи могут быть не симметричными: один из них (на компьютере *A*) обеспечивает *передачу* данных из ОП в линию связи, а другой — *прием* данных из линии связи в буфер ОП.

Механизм передачи байта из компьютера *A* в компьютер *B* аналогичен механизму взаимодействия компьютера с принтером. На стороне компьютера *A* приложение размещает передаваемые данные в буфер ОП, драйвер MMC загружает байт из буфера ОП в буфер ИК, после чего инициируется работа ИК, которая последовательно передает биты в линию связи, дополняя каждый новый байт стартовым и стоповым битами.

На стороне компьютера *B* ИК принимает биты, поступающие со стороны внешнего интерфейса, и помещает их в буфер ИК. 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 MMC, который должен быть запущен еще до начала передачи информации из компьютера *A*. Драйвер MMC переписывает принятый байт из буфера ИК в заранее зарезервированный буфер ОП компьютера *B*. Функциональность драйверов MMC легко расширить так, чтобы каждый из них мог поддерживать как передачу, так и прием данных.

Связав электрически и информационно два автономно работающих компьютера, мы получили простейшую **компьютерную сеть**.

ПРИМЕЧАНИЕ

В «настоящих» локальных сетях подобные функции передачи данных в линию связи выполняются сетевыми интерфейсными картами (Network Interface Card, NIC), называемыми также сетевыми адаптерами, и их драйверами.

Итак, мы имеем в своем распоряжении инструмент, который позволяет приложениям, выполняющимся на разных компьютерах, обмениваться данными.

И хотя приложение на компьютере *A* по-прежнему не может управлять принтером, подключенным к компьютеру *B*, оно может теперь воспользоваться средствами ММС, чтобы передать приложению на компьютере *B* «просьбу» выполнить требуемую операцию. Эти «просьбы» выражаются в виде **сообщений**, передаваемых по каналу ММС между компьютерами. Приложение на компьютере *A* должно «объяснить» приложению на компьютере *B*, какую операцию необходимо выполнить, с какими данными, на каком из имеющихся в его распоряжении устройствах, в каком виде должен быть распечатан текст и т. п. Обо всем этом приложения на компьютерах *A* и *B* договариваются путем обмена сообщениями.

Чтобы приложения могли «понимать» получаемую друг от друга информацию, программисты, разрабатывающие эти приложения, должны *строго оговорить* форматы сообщений, которыми будут обмениваться приложения, и их семантику. Например, они могут договориться о том, что любое выполнение удаленной операции печати начинается с передачи сообщения, запрашивающего информацию о готовности приложения на компьютере *B*; что в следующем сообщении идут идентификаторы компьютера и пользователя, сделавшего запрос; что признаком срочного завершения печати является определенная кодовая комбинация и т. п. Тем самым определяется **протокол взаимодействия приложений**.

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

А теперь посмотрим, как работают вместе все элементы этой простейшей компьютерной сети при решении задачи совместного использования принтера (рис. 1.7).

1. В соответствии с принятым протоколом приложение на компьютере *A* формирует сообщение-запрос к приложению на компьютере *B* и помещает его в буфер ОП. Чтобы передать данное сообщение через канал межмашинной связи приложение на компьютере *A* обращается к ОС, снабжая ее необходимой информацией.
2. ОС запускает драйвер ММС, сообщая ему адрес буфера ОП, где хранится сообщение.
3. Драйвер ММС и интерфейсная карта компьютера *A*, взаимодействуя с драйвером и интерфейсной картой компьютера *B*, передают сообщение байт за байтом в буфер ОП компьютера *B*.
4. Приложение на компьютере *B* извлекает сообщение из буфера, интерпретирует его и выполняет соответствующие действия. В число таких действий входят обращения к ОС с запросами на выполнение тех или иных действий с локальным принтером.

5. ОС запускает драйвер принтера, который в кооперации с интерфейсной картой и контроллером принтера выполняет требуемые операции печати.

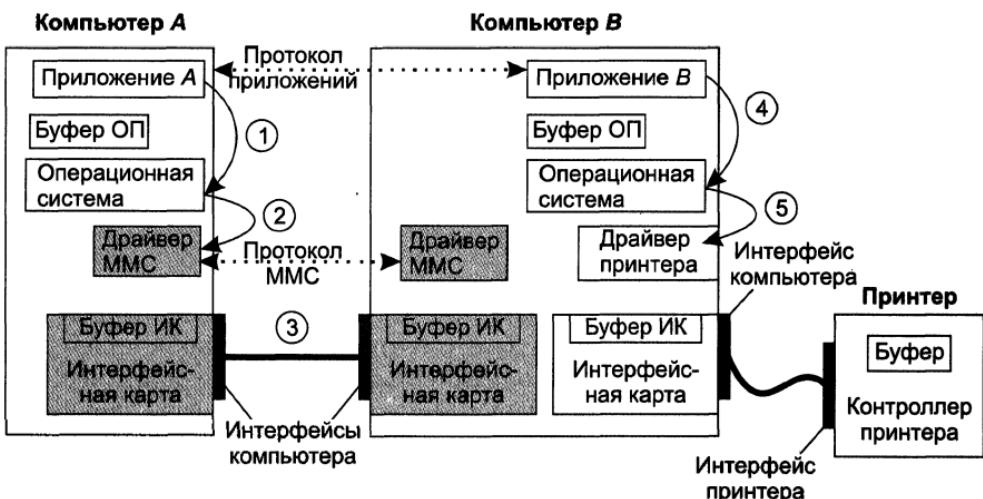


Рис. 1.7. Совместное использование принтера в компьютерной сети

В ходе печати могут возникнуть ситуации, о которых приложение на компьютере *B* должно оповестить приложение на компьютере *A*, например, об отсутствии бумаги в принтере. Для передачи такого сообщения используется симметричная схема: теперь запрос на передачу сообщения поступает от приложения на компьютере *B* к локальной ОС компьютера *B*. ОС запускает драйвер MMC инициирующий побайтную передачу сообщения, которое затем помещается в буфер приложения на компьютере *A*.

Передача данных по линиям связи

Даже при рассмотрении простейшего случая соединения двух компьютеров мы столкнулись с несколькими проблемами физической передачи сигналов по линиям связи в компьютерной сети.

В компьютерных сетях информация передается по линиям связи с помощью электрического тока или напряжения, радиосигналов или световых сигналов – все эти физические процессы представляют собой колебания электромагнитного поля различной частоты.

Линии связи для распространения этих электромагнитных колебаний используют разные виды **физической среды передачи данных**. Это может быть, например, набор проводников, по которым передаются сигналы. На основе *проводных сред* строятся кабельные линии связи.

Кабельные линии связи имеют достаточно сложную конструкцию. Кабель состоит из проводников (физической среды), заключенных в несколько слоев изоляции: электрической, электромагнитной, механической и, возможно,

климатической. Кроме того, кабель может быть оснащен разъемами, позволяющими быстро присоединять к нему различное оборудование.

В качестве физической среды также используется земная атмосфера или космическое пространство, через которое распространяются информационные сигналы в виде различных видов электромагнитных волн: радиоволны, сигналы сверхвысокой частоты, инфракрасные волны и лазерные лучи. В этом случае говорят о *беспроводной среде*, основой которой являются беспроводные линии связи. Они используются чаще всего в тех случаях, когда кабельные линии связи применить нельзя, например, при прохождении линии связи через малонаселенную местность или же для связи с мобильными пользователями.

В вычислительной технике для представления данных используется **двоичный код**. Линии связи предоставляют только потенциальную возможность передачи дискретной информации. Для того чтобы передатчик и приемник, соединенные линией связи, могли обмениваться информацией, им необходимо договориться о том, какие сигналы будут соответствовать двоичным единицам и нулям дискретной информации. Для представления дискретной информации в среде передачи данных применяются сигналы двух типов: прямоугольные импульсы (рис. 1.8, *a*) и синусоидальные волны (рис. 1.8, *б*, *в* и *г*). В первом случае используют термин **кодирование**, во втором — **модуляция**.

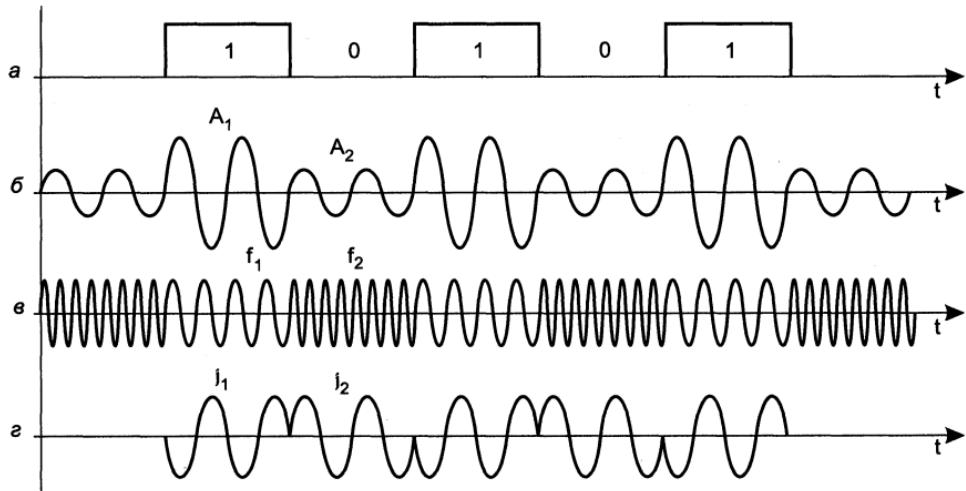


Рис. 1.8. Различные типы модуляции

Для представления нулей и единиц посредством модуляции устройство-преобразователь меняет параметры **несущего** синусоидального сигнала. В зависимости от того, какой параметр синусоиды изменяется, различают следующие основные виды модуляции:

- **амплитудная модуляция** (Amplitude Shift Keying, ASK), при которой для логической единицы выбирается один уровень амплитуды синусоиды несущей частоты, а для логического нуля — другой;

- **частотная модуляция** (Frequency Shift Keying, **FSK**), при которой значения нуля и единицы исходных данных передаются синусоидами с различной частотой — f_0 и f_1 ;
- **фазовая модуляция** (Phase Shift Keying, **PSK**), при которой значениям данных 0 и 1 соответствуют сигналы одинаковой частоты, но различной фазы, например 0 и 180° или 0, 90, 180 и 270°.

Для снижения стоимости линий связи в компьютерных сетях обычно стремятся к сокращению количества проводов и из-за этого передают все биты одного байта или даже нескольких байтов не параллельно, как внутри компьютера, а *последовательно*, побитно передавая байты всего по одной паре проводов.

Еще одной проблемой, которую нужно решать при передаче сигналов по линиям связи, 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), существует вероятность искажения некоторых битов передаваемых данных. Для повышения надежности передачи данных между компьютерами часто используется стандартный прием — подсчет **контрольной суммы** и передача ее по линиям связи после каждого байта или после некоторого блока байтов.

В зависимости от того, могут ли линии связи передавать информацию в обоих направлениях или нет, они делятся на следующие типы.

- **Симплексная линия связи** позволяет передавать информацию только в одном направлении.
- **Дуплексная линия связи** обеспечивает одновременную передачу информации в обоих направлениях. Дуплексная линия связи 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, в этом случае применяют дополнительные методы выделения каждого потока из суммарного сигнала.
- **Полудуплексная линия связи** 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.

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. 1.9), функции которых ранее выполнялись приложениями на компьютерах *A* и *B* соответственно. Теперь эта пара клиент–сервер может быть использована любым приложением, выполняемым на компьютере *A*.

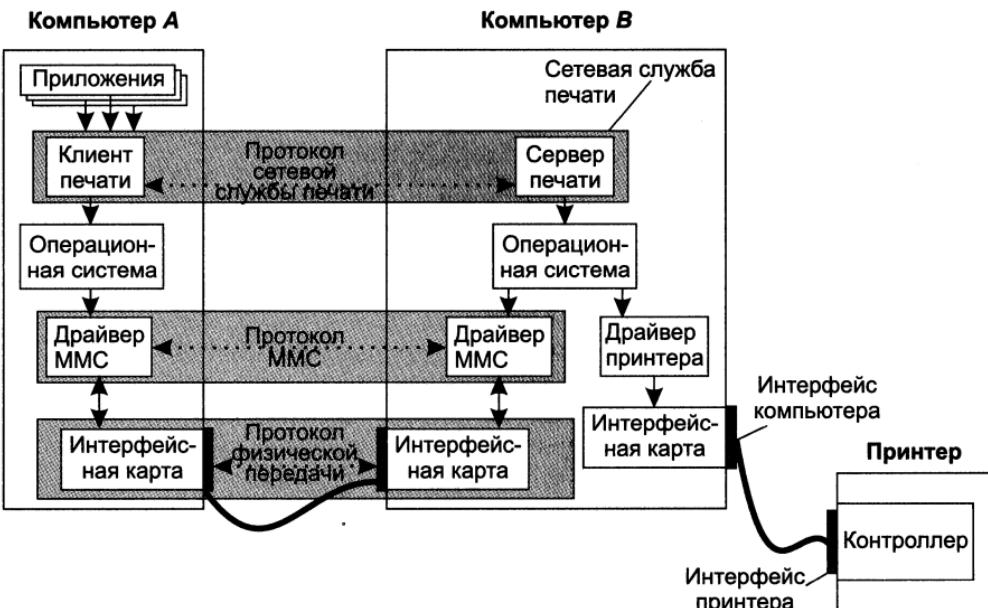


Рис. 1.9. Совместное использование принтера в компьютерной сети с помощью сетевой службы печати

Обобщая такой подход применительно к другим типам разделяемых ресурсов, дадим следующие определения¹.

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 обработать, 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. 1.9 модули клиента и сервера, реализующие удаленный доступ к принтеру, образуют **сетевую службу печати**.

Файловая служба позволяет получать доступ к файлам, хранящимся на диске других компьютеров. Серверный компонент файловой службы называют **файл-сервером**.

Возвращаясь к схеме веб-серфинга в Интернете (см. рис. 1.4), мы видим, что веб-браузер (названный нами уже тогда клиентом) вместе с веб-сервером образуют сетевую **веб-службу**. 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. 1.10, два компьютера связаны не непосредственно, как это было во всех предыдущих примерах, а через Интернет. Для того чтобы отразить этот факт графически, мы поместили между двумя компьютерами так называемое **коммуникационное облако**, которое позволяет нам 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 эти сообщения «из рук в руки» (от интерфейса одного компьютера к интерфейсу другого) или через большое число посредников – промежуточных коммуникационных устройств. Вместе с тем усложнение среды передачи сообщений приводит к возникновению новых дополнительных задач, на решение которых не был рассчитан рассмотренный ранее драйвер

¹ Термины «клиент» и «сервер» являются чрезвычайно многозначными. Данная пара терминов, уже использованная нами для обозначения функциональной роли взаимодействующих компьютеров и приложений, применима также к программным модулям.

MMC. Вместо него на взаимодействующих компьютерах должны быть установлены более развитые транспортные средства.

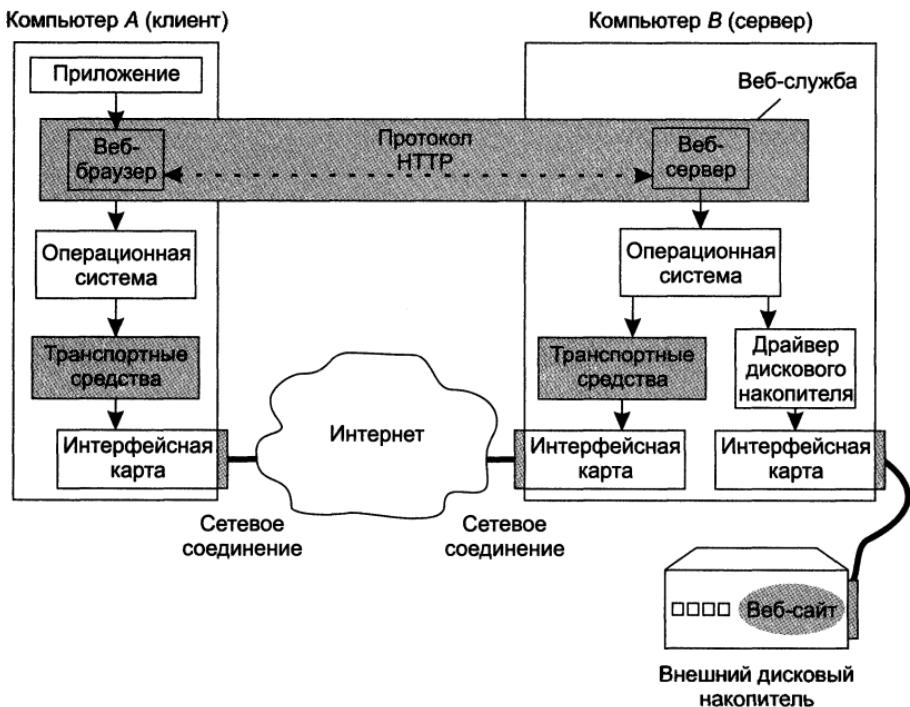


Рис. 1.10. Веб-служба

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. 1.9 и 1.10), мы видим, что удаленный доступ к сетевым ресурсам обеспечивается:

- сетевыми службами;
- средствами транспортировки сообщений по сети — транспортными средствами.

Следовательно, именно эти функциональные модули должны быть добавлены к ОС, чтобы она могла называться сетевой (рис. 1.11).

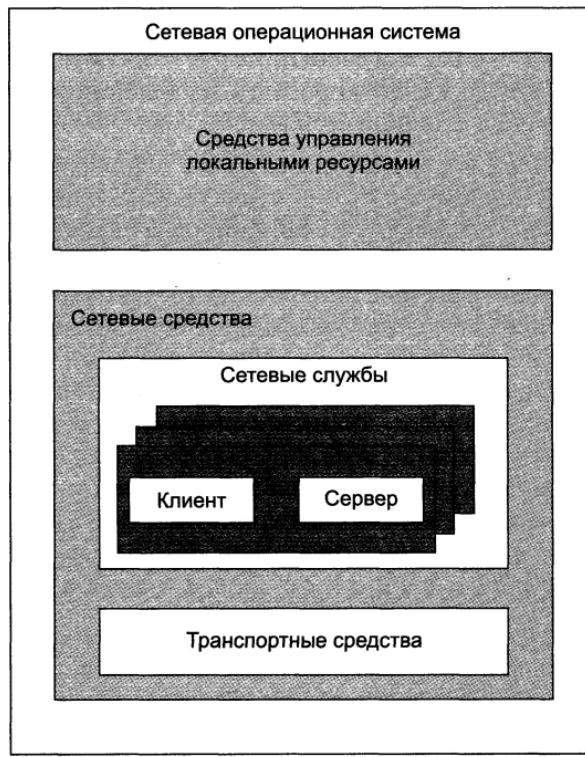


Рис. 1.11. Функциональные компоненты сетевой ОС

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, **централизованная справочная служба**, или **служба каталогов**, предназначена для ведения базы данных о пользователях сети, обо всех ее программных и аппаратных компонентах¹. Другими примерами являются **служба мониторинга сети**, позволяющая захватывать и анализировать сетевой трафик, **служба безопасности**, в функции которой может входить, в частности, выполнение процедуры логического входа с проверкой пароля, **служба резервного копирования и архивирования**.

¹ Например, служба каталогов Active Directory компании Microsoft.

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 должна включать *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 приобретается отдельно. 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, драйверов и протокольных модулей принимается разработчиками с учетом самых разных соображений: технических, коммерческих и даже юридических. Так, например, именно на основании антимонопольного закона США компания 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 применять файлы друг друга. Компьютеры, совмещающие функции клиента и сервера, называют одноранговыми узлами.

Операционная система, которая преимущественно содержит клиентские части сетевых служб, называется **клиентской**. Клиентские ОС устанавливаются на компьютеры, обращающиеся с запросами к ресурсам других компьютеров сети. За такими компьютерами, так же 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.

Сетевые приложения

Компьютер, подключенный к сети, может выполнять следующие типы приложений.

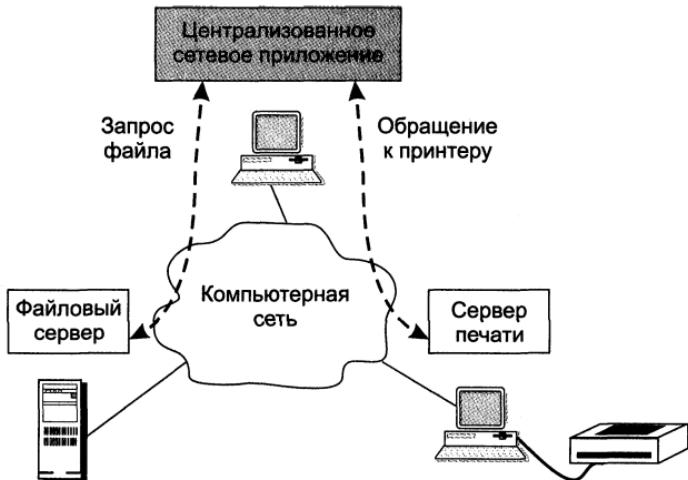
- **Локальное приложение** целиком выполняется на данном компьютере и использует только локальные ресурсы (рис. 1.12, а). Для такого приложения не требуется никаких сетевых средств, оно может быть выполнено на автономно работающем компьютере.
- **Централизованное сетевое приложение** целиком выполняется на данном компьютере, но обращается в процессе своего выполнения к ресурсам других компьютеров сети. В примере на рис. 1.12, б приложение, которое выполняется на клиентском компьютере, обрабатывает данные из файла, хранящегося на файловом 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. 1.12, 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, предназначенному, скажем, для анализа климатических изменений, можно выделить три достаточно самостоятельные части (см. рис. 1.12, 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, а 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 часть 1, ответственную за поддержку специализированного пользовательского интерфейса, можно было бы запустить на нескольких персональных компьютерах, что позволило бы работать с этим приложением некоторым пользователям одновременно.

а



б



в

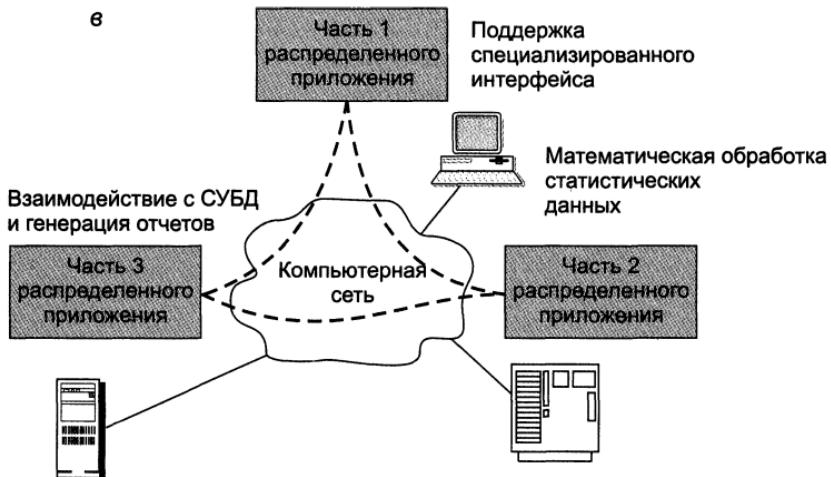


Рис. 1.12. Типы приложений, выполняющихся в сети

Однако чтобы добиться всех тех преимуществ, которые сулят распределенные приложения, разработчикам этих приложений 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 т. д., 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. 1.13, 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.

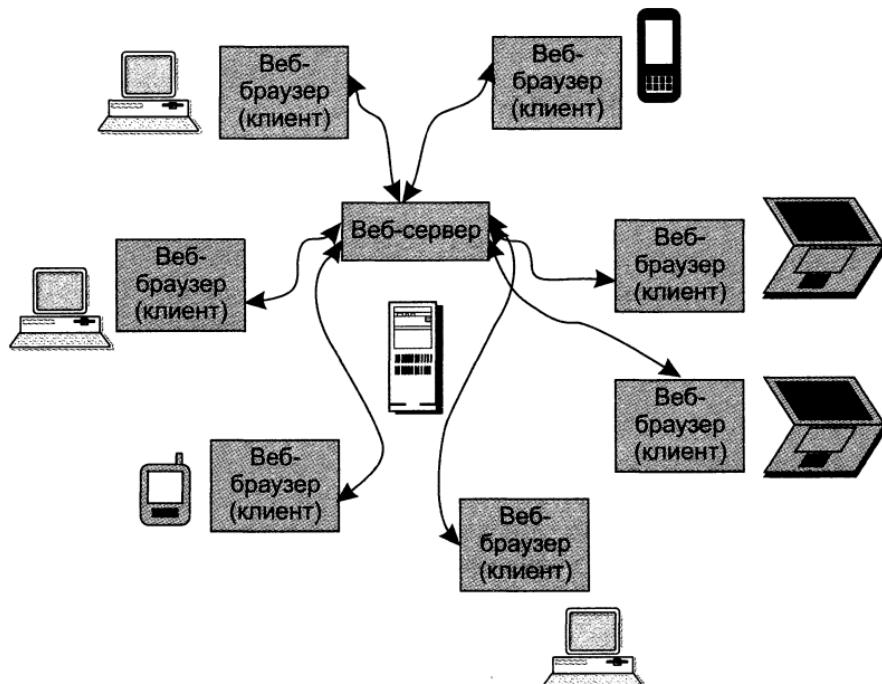


Рис. 1.13. Веб-служба как распределенное приложение

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 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 последних и очень 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.

Проблемы связи нескольких компьютеров

До сих пор мы ограничивались обсуждением средств передачи сообщений в вырожденной сети, состоящей всего из двух машин. В этом разделе мы усложним задачу и выясним, какие принципиально новые проблемы возникают в сети, если она объединяет больше двух компьютеров.

Топология физических связей

Как только число компьютеров становится больше двух, мы сталкиваемся с проблемой выбора топологии сети.

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.

Количество возможных конфигураций резко возрастает при увеличении количества связываемых устройств. Так, если три компьютера мы можем связать двумя способами (рис. 1.14, *a*), то для четырех можно предложить уже шесть топологически разных конфигураций (при условии неразличимости компьютеров), что и иллюстрирует рис. 1.14, *b*.

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.

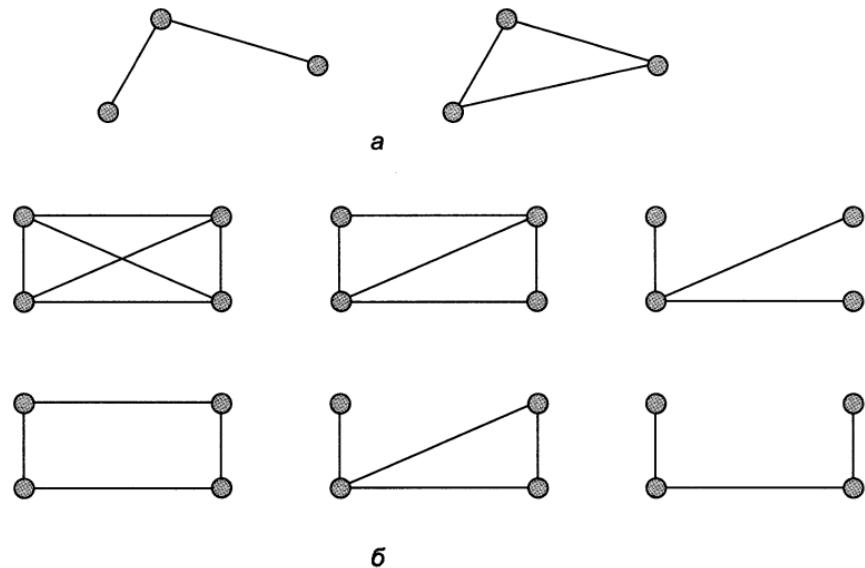


Рис. 1.14. Варианты связи компьютеров

От выбора топологии связей существенно зависят характеристики сети. Например, наличие между узлами нескольких путей повышает надежность сети и делает возможным распределение нагрузки 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 (рис. 1.15, а) соответствует сети, в которой каждый компьютер непосредственно связан со всеми остальными. Несмотря на логическую простоту, этот вариант оказывается громоздким и неэффективным. Действительно, в таком случае каждый компьютер в сети должен иметь большое количество коммуникационных портов (сетевых интерфейсов), достаточное для связи с каждым из всех остальных компьютеров сети. Для каждой пары компьютеров должна быть выделена отдельная физическая линия связи¹. Поэтому полносвязные топологии применяются в сетях, объединяющих небольшое количество компьютеров.

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

¹ В некоторых случаях даже две линии, если использование одной линии для двусторонней передачи невозможно.

Ячеистая топология¹ получается из полносвязной путем удаления некоторых связей (рис. 1.15, б). Ячеистая топология допускает соединение большого количества компьютеров и характерна, как правило, для крупных сетей.

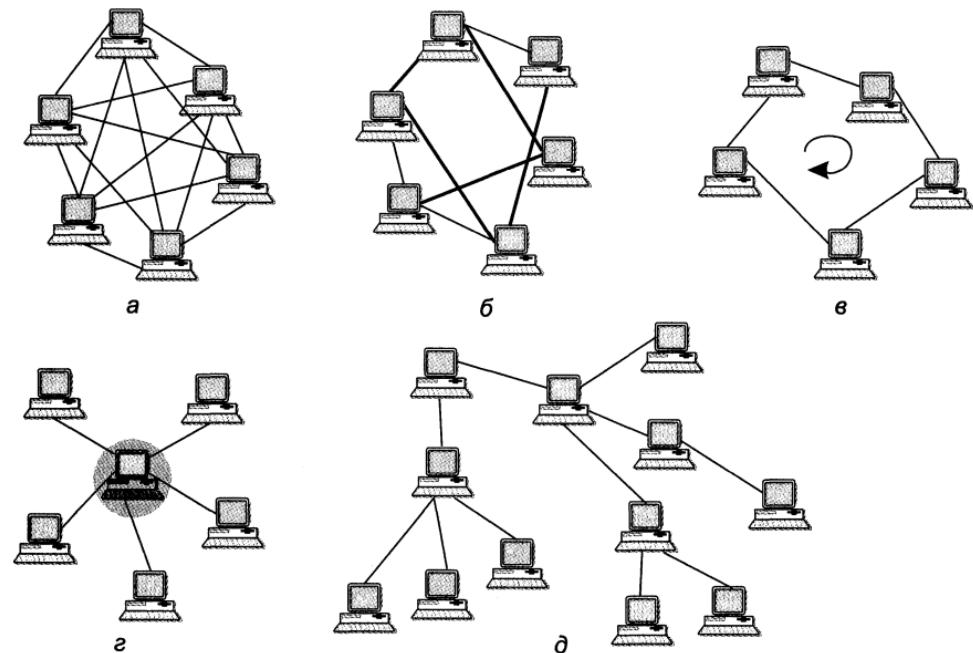


Рис. 1.15. Типовые топологии сетей

В сетях с **кольцевой топологией** (рис. 1.15, 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. Кольцо представляет собой очень удобную конфигурацию и 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. 1.15, г) образуется в случае, когда каждый компьютер подключается непосредственно к общему центральному

¹ Иногда ячеистой называют полносвязную или близкую к полносвязной топологию.

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 топологии типа «звезда» 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 связями типа «звезда» (рис. 1.15, 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.

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 фрагменты (подсети), имеющие типовую топологию, поэтому их называют сетями со **смешанной топологией** (рис. 1.16).

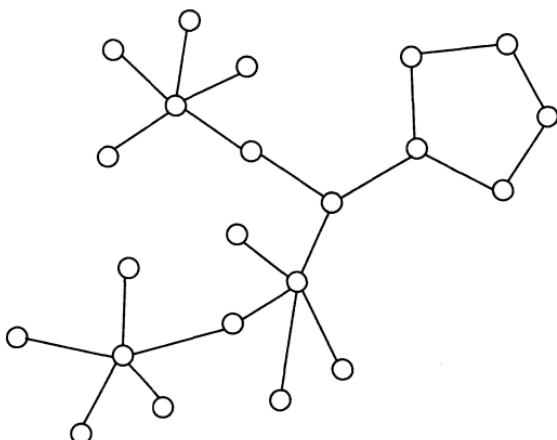


Рис. 1.16. Смешанная топология

Адресация узлов сети

Еще одной новой проблемой, которую нужно учитывать при объединении трех и более компьютеров, является проблема их адресации, точнее адресации

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

их сетевых интерфейсов¹. 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:

- **的独特地址 (unicast)** используется для идентификации отдельных интерфейсов;
- **组播地址 (multicast)** идентифицирует сразу несколько интерфейсов, поэтому данные, помеченные групповым адресом, доставляются каждому из интерфейсов (узлов), входящих в группу;
- **广播地址 (broadcast)** идентифицирует адреса всех сетевых интерфейсов;
- **任意播地址 (anycast)**, так же как и групповой адрес, задает группу адресов, однако данные, посланные по этому адресу, должны быть доставлены не всем адресам данной группы, а любому из них.

На рис. 1.17 показано несколько примеров использования разных типов адресов в полносвязной сети, в которой каждый компьютер оснащен парой интерфейсов. Здесь числа 1, 2, ..., 6 являются уникальными адресами, однозначно идентифицирующими сетевые интерфейсы. Данные, снабженные уникальным адресом 3, должны быть доставлены одному интерфейсу, имеющему уникальный адрес 3. Пусть в сети, помимо уникальных адресов, определен групповой адрес 55. Этот адрес присвоен двум интерфейсам с уникальными адресами 4 и 5. Данные, направленные по этому адресу, будут переданы сразу на эти оба интерфейса. И наконец, если данным приписан широковещательный адрес — в нашем примере для данного типа адреса зарезервирован код 111 — то они поступят на все сетевые интерфейсы сети.

Адреса могут быть **числовыми**, как, например, в предыдущем примере, и **символьными**. Символьные адреса (имена) предназначены для запоминания людьми и поэтому обычно несут смысловую нагрузку, например, www.gazeta.ru или biletvteatr.ru. Хотя символьные имена удобны для людей, из-за переменного формата и потенциально большой длины для передачи по сети в основном используются числовые адреса, например, 0081005e24a8, 129.26.255.255, 81.1a.ff.ff.

Множество всех адресов, которые являются допустимыми в рамках некоторой схемы адресации, называется адресным пространством.

Адресное пространство может иметь плоскую или иерархическую организацию.

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

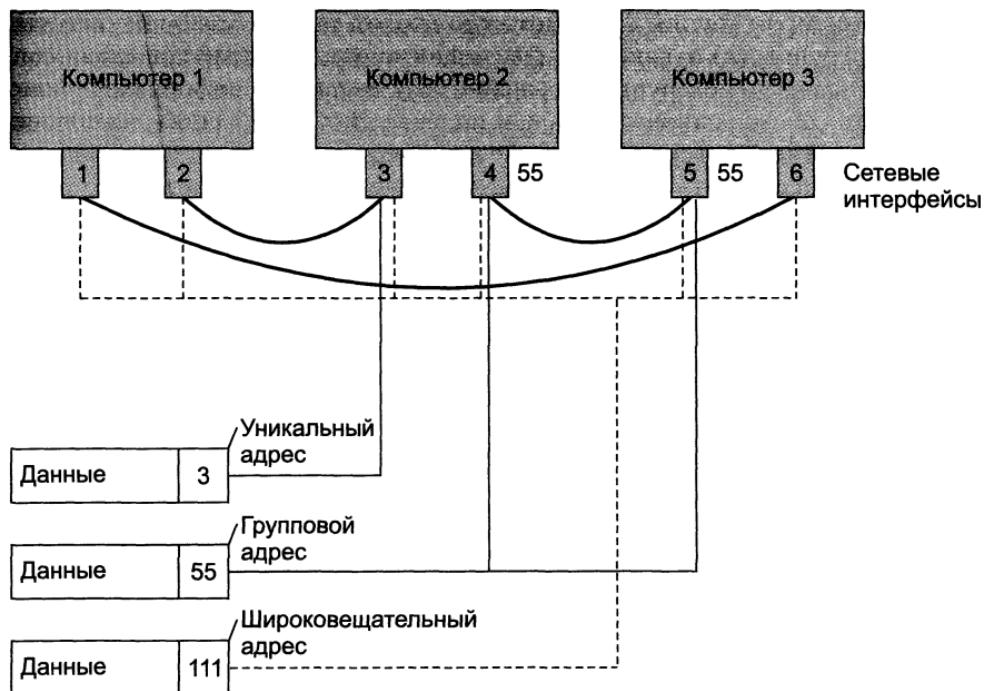


Рис. 1.17. Уникальные, групповые и широковещательные адреса

При **плоской** организации множество адресов никак не структурировано. Пусть, например, компьютерная сеть поликлиники состоит из 50 компьютеров, по 10 компьютеров на каждом из 5 этажей. Можно просто последовательно пронумеровать все компьютеры от 1 до 50 или присвоить им подобные символьные имена, например, comp1, comp2, ..., comp50. А можно было бы выбрать другой вариант — закрепить за компьютерами произвольные неповторяющиеся числа или использовать для адресации набор из 50 неповторяющихся символьных последовательностей. Во всех этих случаях получается плоская адресация.

При **иерархической** организации адресное пространство организовано в виде вложенных друг в друга подгрупп, которые, последовательно сужая адресуемую область, в конце концов, определяют отдельный сетевой интерфейс. Обратимся снова к адресации компьютеров в сети поликлиники. Пусть, например, вы решили использовать для этой цели адреса, состоящие из двух частей, одна из которых обозначает этаж, а вторая идентифицирует компьютер в пределах этажа. Числовые адреса в таком случае могли бы иметь вид 1-1, 1-2, ..., 5-10, а символьные — (store1, comp1), (store1, comp2), ..., (store5, comp10). Здесь применена двухуровневая иерархическая адресация (числовая или символьная).

В показанной на рис. 1.18 трехуровневой структуре адресного пространства адрес конечного узла (g , s , n) задается тремя составляющими:

идентификатором группы (g), в которую входит данный узел, идентификатором подгруппы (s) и, наконец, 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 (g), затем для дальнейшей локализации адресата задействовать следующую по старшинству часть (s) и в конечном счете — младшую часть (n).

(g, s, n) — трехуровневый иерархический адрес

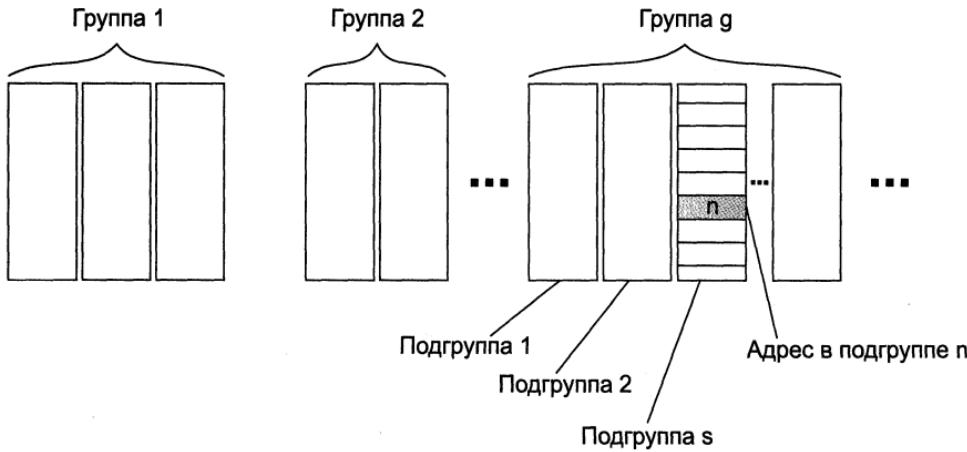


Рис. 1.18. Иерархическая организация адресного пространства

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

Адреса могут назначаться интерфейсам в результате выполнения программной процедуры конфигурирования либо встраиваться в аппаратуру компанией-изготовителем. В последнем случае адреса называют **аппаратными** (hardware addresses). Использование аппаратных адресов является жестким решением — при замене аппаратуры, например сетевого адаптера, изменяется и адрес сетевого интерфейса компьютера.

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, в которой соответствующий вид адресации наиболее удобен. А для преобразования адресов из одного вида в другой применяют специальные вспомогательные процедуры, которые называют **протоколами разрешения адресов**.

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 UDP¹.

Коммутация

Мы уже отмечали, что в сети с неполносвязной топологией обмен данными между любой произвольной парой узлов в общем случае идет через транзитные узлы. Рассмотрим, например, сеть на рис. 1.19. Здесь узлы сети пронумерованы цифрами, а интерфейсы идентифицируются буквами. 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*.

Последовательность узлов, лежащих на пути от отправителя к получателю, образует **маршрут**.

В данном примере маршрутом является последовательность: 2—1—5—4, где 2 — узел-отправитель, 1 и 5 — транзитные узлы, 4 — узел-получатель.

В общем случае один и тот же сетевой узел может выступать в роли отправителя, получателя и транзитного узла.

Соединение отправителя и получателя через сеть транзитных узлов называют **коммутацией**.

Для выполнения коммутации должны быть решены следующие задачи:

- определение потоков данных;
- определение маршрутов;

¹ О номерах портов TCP и UDP читайте в главе 3.

- продвижение данных в каждом транзитном узле;
- мультиплексирование и демультиплексирование потоков.

Определение потоков данных

Понятно, что через один транзитный узел может проходить несколько маршрутов, например, через узел 5 (см. рис. 1.19) проходят все маршруты, по которым узлы 3, 4 и 10 обмениваются данными с другими узлами сети. Транзитный узел должен уметь *распознавать* поступающие на него потоки данных, для того чтобы обеспечивать передачу каждого из них именно на тот свой интерфейс, который ведет к нужному узлу.

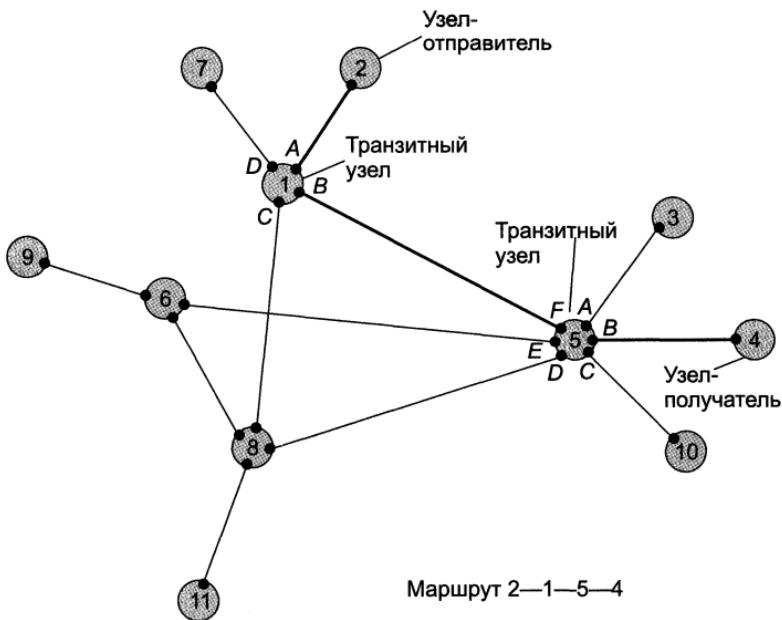


Рис. 1.19. Коммутация через сеть транзитных узлов

Информационным потоком, или потоком данных, называют передаваемые по сети данные, объединенные набором общих признаков, выделяющих их из общего сетевого трафика. Для каждого из потоков может быть проложен свой особый маршрут.

При коммутации в качестве *обязательного* признака выступает адрес назначения данных. На основании этого признака все данные, поступающие в транзитный узел, разделяются на потоки, и каждый поток передается на тот интерфейс, через который пролегает маршрут к соответствующему узлу назначения.

Адрес источника в совокупности с адресом назначения определяют информационный поток для этой пары узлов.

В качестве признаков потока могут также выступать идентификаторы приложений, генерирующих эти данные. 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 для генерируемых приложением данных должен осуществляться с учетом их характера. Например, для файлового сервера важно, чтобы передаваемые им большие объемы данных направлялись по каналам, обладающим высокой пропускной способностью, а для программной системы управления, которая посыпает в сеть короткие сообщения, требующие обязательной и немедленной отработки, при выборе маршрута более важна надежность линии связи и минимальный уровень задержек на маршруте.

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 адресов конечных узлов для идентификации потока – 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. 1.19, узел 1 может быть настроен так, чтобы передавать на интерфейс *B* все данные, поступившие с интерфейса *A*, а на интерфейс *C* – все 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 несет все данные потока. **Глобальная метка** назначается данным потоком 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.

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, идентификаторы интерфейсов, метки информационных потоков.

Определение маршрутов

Определить маршрут — это значит 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, время или надежность доставки данных получателю по выбранному маршруту. На практике для снижения объема вычислений ограничиваются поиском не оптимального в математическом смысле, а рационального, то есть близкого к оптимальному маршрута.

Задача выбора еще более упрощается за счет того, что при анализе критерия оптимизации учитываются далеко не все факторы, влияющие на этот критерий. Пусть, например, мы ищем самый лучший с точки зрения времени доставки маршрут передачи данных от узла 2 к узлу 4 (рис. 1.20). На времени доставки, очевидно, сказываются топология сети (количество транзитных узлов, которые должны пройти данные), скоростные характеристики каналов связи, их загруженность, надежность каналов и транзитных устройств, а также многие другие факторы.

Из рисунка видно, что для передачи трафика между узлами 2 и 4 существует два альтернативных маршрута: 2–1–5–4 и 2–1–8–5–4. Если мы учтем только топологию, то выбор очевиден — маршрут 2–1–5–4, который имеет меньше транзитных узлов.

Однако если принять во внимание скоростные характеристики каналов, то наше решение может измениться. Действительно, как показано на рисунке, каналы 1–8 и 8–5 характеризуются скоростью передачи данных 100 Мбит/с (мегабит в секунду), а канал 1–8 — только 10 Мбит/с. Если мы хотим, чтобы данные поступали к получателю как можно быстрее, то нам следовало бы выбрать маршрут 2–1–8–5–4, хотя он и проходит через большее количество промежуточных узлов.

В обоих случаях, выбирая маршрут, мы не учитывали текущую степень загруженности каналов трафиком. Используя аналогию с автомобильным трафиком, можно сказать, что мы выбирали маршрут по карте, учитывая количество промежуточных городов (аналог количества транзитных узлов) и отдавая предпочтение магистралям (что соответствует выбору скоростных каналов). Но мы не стали слушать радио, информирующее о текущих заторах на дорогах. А это могло бы оказаться решающим образом на качестве нашего маршрута.

Задачи поиска и выбора маршрутов можно решать «вручную» или автоматически. В первом случае администратор сети определяет маршрут эмпирически 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.

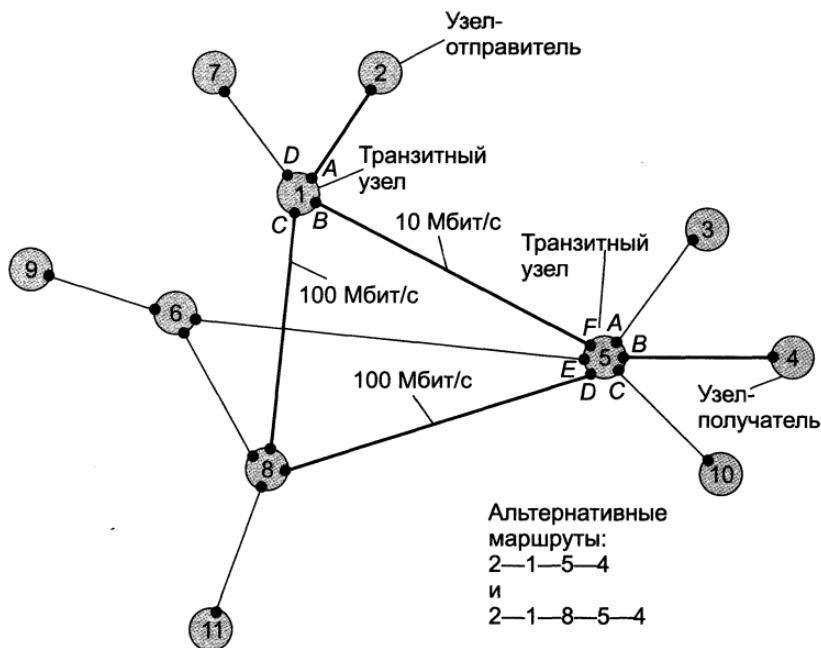


Рис. 1.20. Выбор маршрута

Однако эмпирический подход к определению маршрутов мало пригоден для большой сети со сложной топологией. В этом случае применяют автоматические методы определения маршрутов. Для этого сетевые устройства оснащаются специальными программными средствами, которые организуют взаимный обмен информацией о конфигурации связей каждого отдельного узла. На основе собранных данных программными методами воспроизводится топология сети и определяются рациональные маршруты.

После того как маршрут определен (вручную или автоматически), его надо *проложить* в сети, то есть настроить транзитные узлы так, чтобы они передавали данные в соответствии с выбранным маршрутом. Администратор сети может зафиксировать маршрут, выполнив в ручном режиме конфигурирование устройства, например, жестко скоммутировав на длительное время определенные пары входных и выходных интерфейсов (так работали «телефонные барышни» на первых коммутаторах).

Более гибким решением является создание **таблиц коммутации**, хранящихся в памяти каждого из транзитных узлов. Таблица коммутации состоит из записей, в которых признакам потоков (например, адресам назначения или номерам входных интерфейсов) ставятся в соответствие номера интерфейсов,

на которые устройство должно передать эти потоки. Например, запись (N, B) означает следующее: «Каждый раз, когда в устройство поступят данные, относящиеся к потоку N , их следует передать для дальнейшего продвижения на интерфейс B ».

Записи в таблицу коммутации могут заноситься вручную администратором сети. Однако поскольку топология и состав информационных потоков может меняться (отказы узлов или появление новых промежуточных узлов, изменение адресов или определение новых потоков), гибкое решение задач определения и задания маршрутов предполагает постоянный анализ состояния сети, обновление маршрутов, а следовательно, и таблиц коммутации. В таких случаях и задача поиска маршрутов, и задача формирования таблиц коммутации не могут быть решены без использования на всех транзитных узлах достаточно сложных программных и аппаратных средств.

Продвижение данных

Итак, пусть маршруты определены, записи о них занесены в таблицы коммутации, все готово к передаче данных между узлом-отправителем и узлом-получателем. Для каждой пары взаимодействующих узлов эта операция может быть представлена несколькими локальными операциями коммутации: каждый отдельный транзитный узел должен соответствующим образом выполнить «переброску» данных с одного своего интерфейса на другой, другими словами, выполнить **коммутацию интерфейсов**. Устройство, функциональным назначением которого является коммутация, называется **коммутатором**. Таким образом, все транзитные узлы представляют собой коммутаторы.

ПРИМЕЧАНИЕ

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 быть выполнена в соответствии с различными правилами и алгоритмами. Некоторые способы коммутации и соответствующие им таблицы и устройства получили специальные названия. Например, в технологиях сетевого уровня, таких как IP и IPX, 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», используемые в узком смысле, например, в смысле коммутации и коммутатора локальной сети. Для телефонных сетей, которые появились намного раньше компьютерных, так же характерна аналогичная терминология, коммутатор является здесь синонимом телефонной станции.

На рис. 1.21 приведен упрощенный алгоритм работы коммутатора. Прежде чем выполнить коммутацию интерфейсов, коммутатор должен распознать поток, то есть выделить признак потока — пусть им будет N . Затем этот признак сравнивается с каждым из признаков, заданных в таблице коммутации. Если произошло совпадение, то из соответствующей строки таблицы определяется идентификатор интерфейса (в данном случае — B), на который и направляется поток N .

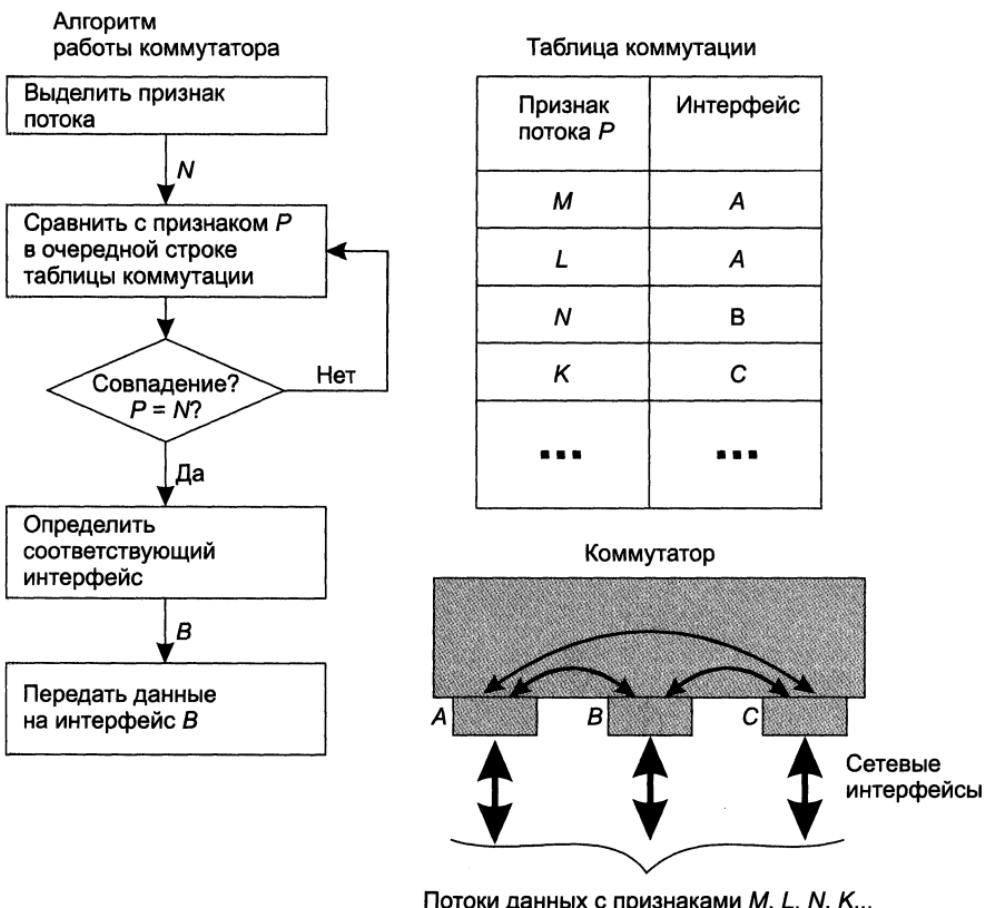


Рис. 1.21. Алгоритм работы коммутатора

А теперь посмотрим, как работает коммутатор на примере уже знакомой нам сети (см. рис. 1.20). На рис. 1.22 приведена коммутационная таблица транзитного узла 1. В качестве признака потока здесь используется адрес назначения данных.

Таблица построена на основе достаточно очевидного анализа топологии сети: в зависимости от адреса назначения, данные из узла 1 направляются

на тот или иной интерфейс, соответствующий выбранному маршруту. В данном случае при выборе маршрутов минимизировалось число транзитных узлов. Так, в таблице коммутации зафиксировано, что данные, адресованные узлам 3, 4, 5 и 10, должны быть «переброшены» на интерфейс *B*, хотя к этим узлам ведут и другие, более длинные маршруты, пролегающие через интерфейс *C*.

Если к некоторому узлу ведут несколько равноценных маршрутов, например, маршруты *C*—8—6—9 и *B*—5—6—9 к узлу 9, то мы выбираем любой из них. В нашем примере предпочтение отдано первому маршруту.

Коммутатором может быть как специализированное устройство, называемое **аппаратным коммутатором** или просто **коммутатором**, так и универсальный компьютер со встроенным программным механизмом коммутации — **программный коммутатор**.

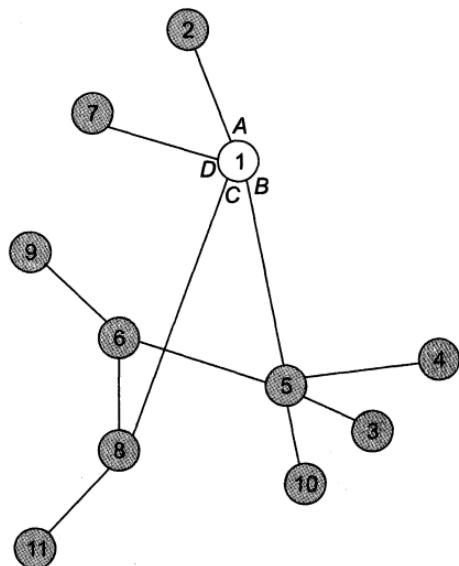


Таблица коммутации узла 1

Признак (адрес)	Интерфейс
2	A
3	B
4	B
5	B
6	C
7	D
8	C
9	C
10	B
11	C

Рис. 1.22. Продвижение данных на основе таблицы коммутации

Транзитные узлы, выполненные на базе универсального компьютера, могут совмещать функции коммутации данных с решением пользовательских задач. Однако во многих случаях более рациональным является решение, в соответствии с которым некоторые узлы в сети выделяются *специально и исключительно* для коммутации. Эти узлы образуют **коммутационную сеть**, к которой подключаются все остальные узлы сети, называемые **конечными**.

На рис. 1.23 показана образованная узлами 1, 5, 6 и 8 коммутационная сеть, к которой подключаются конечные узлы 2, 3, 4, 7, 9, 10 и 11.

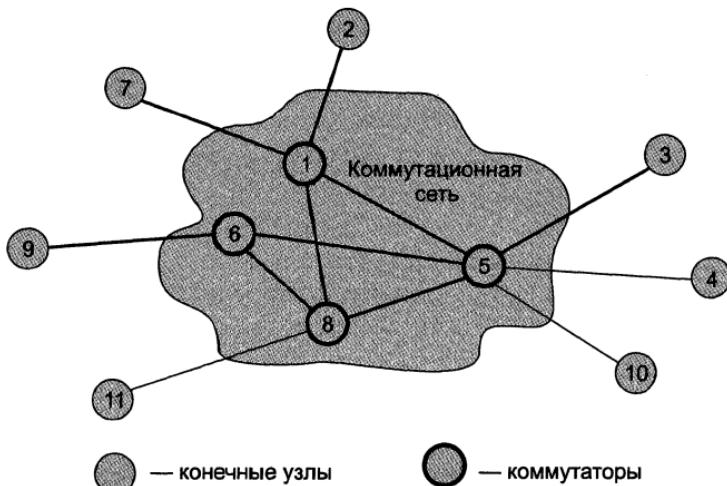


Рис. 1.23. Коммутационная сеть

Пример

Коммутация данных в компьютерной сети во многом схожа с работой традиционной почты. В качестве потоков данных здесь выступают почтовые отправления — письма, посылки и т. п. Почтовые отделения работают подобно коммутаторам, они сортируют почтовые отправления на основе адресов назначения, направляя их в разных слотах, играющие роль интерфейсов коммутатора.

Затем содержимое слотов упаковывают в мешки и погружают в вагоны поездов, автофургоны, морские суда или самолеты, чтобы доставить их на промежуточный почтовый узел для следующей сортировки. Транспортные средства, связывающие почтовые узлы между собой, играют роль физических каналов связи компьютерной сети.

Последовательности промежуточных почтовых отделений (маршруты) для всех направлений заранее определены в результате многолетней работы почтовой службы. Информация о выбранных маршрутах следования почты представлена в каждом почтовом отделении в виде таблицы, в которой задано соответствие между страной, городом, улицей назначения, с одной стороны, и почтовым отделением, в которое надо отправить письмо или посылку, с другой. 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.

Мультиплексирование и демультиплексирование

В неполносвязных сетях, к которым сегодня относятся практически все компьютерные сети, по определению всегда найдется хотя бы пара узлов,

которые не связаны друг с другом непосредственно. Это означает, что поток данных, протекающий между этими узлами, обязательно пройдет как минимум через один транзитный узел и, следовательно, будет «вынужден» разделять линию связи с потоками от других узлов.

Обратимся к сети, изображенной на рис. 1.24. Мы видим, что для обеспечения полной информационной связности большинство линий связи в этой сети должны разделяться между потоками. Так, например, любая линия связи, непосредственно соединяющая конечный узел с коммутационной сетью, должна уметь совмещать передачу нескольких потоков по числу одновременно протекающих сеансов связи этого узла с остальными узлами сети.

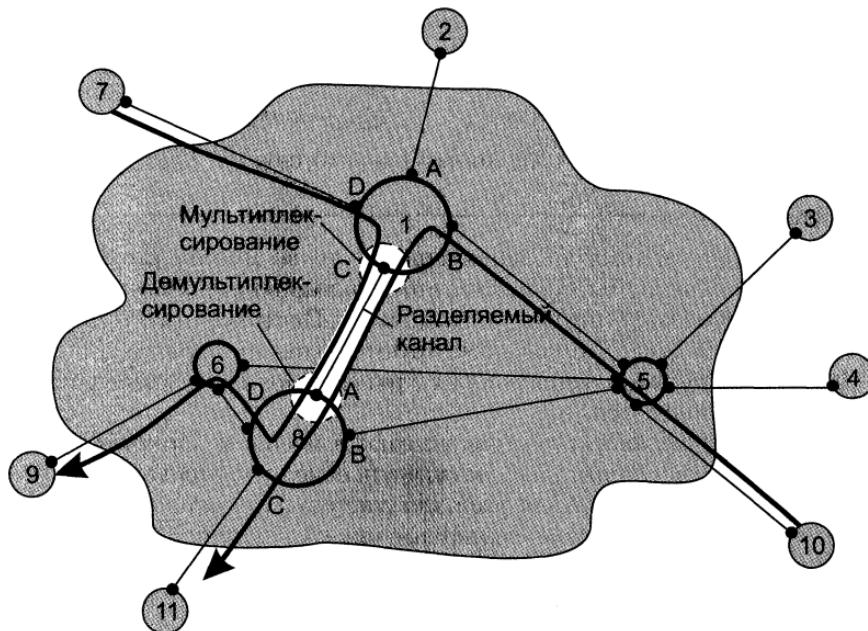


Рис. 1.24. Операции мультиплексирования и демультиплексирования потоков при коммутации

На рисунке показаны два потока: один — это поток данных между узлами 7 и 9, а другой — между узлами 10 и 11. Пусть соответствующие этим потокам сеансы связи проходят одновременно. Маршруты этих потоков таковы, что в некоторый момент времени они оба поступают на коммутатор 1, поток 7—9 поступает на интерфейс D, а поток 10—11 — на интерфейс B. В результате коммутации оба потока «перебрасываются» на один и тот же интерфейс C, а затем направляются в одну и ту же линию связи, соединяющую коммутаторы 1 и 8.

Способ разделения одной линии связи между узлами сети в рамках нескольких одновременно протекающих сеансов связи называется **мультиплексированием**.

При мультиплексировании линии связи из нескольких отдельных потоков образуется общий «смешанный» поток данных, который передается по одной физической среде (например, медному проводу кабельной линии связи). Технология мультиплексирования должна позволять получателю такого суммарного потока (в нашем примере это интерфейс A коммутатора 8) выполнять обратную операцию — **демультиплексирование** (разделение) суммарного потока на слагаемые потоки.

Одним из основных способов мультиплексирования потоков является **разделение времени**. При этом способе каждый поток время от времени (с фиксированным или случайным периодом) получает линию связи 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.

Мультиплексирование и демультиплексирование выполняют коммутаторы.

Частный случай коммутатора, у которого все входящие информационные потоки коммутируются на один выходной интерфейс, где они мультиплексируются в один агрегированный поток, называется **мультиплексором** (рис. 1.25, а).

Коммутатор, который имеет один входной интерфейс и несколько выходных, называется **демультиплексором** (рис. 1.25, б).

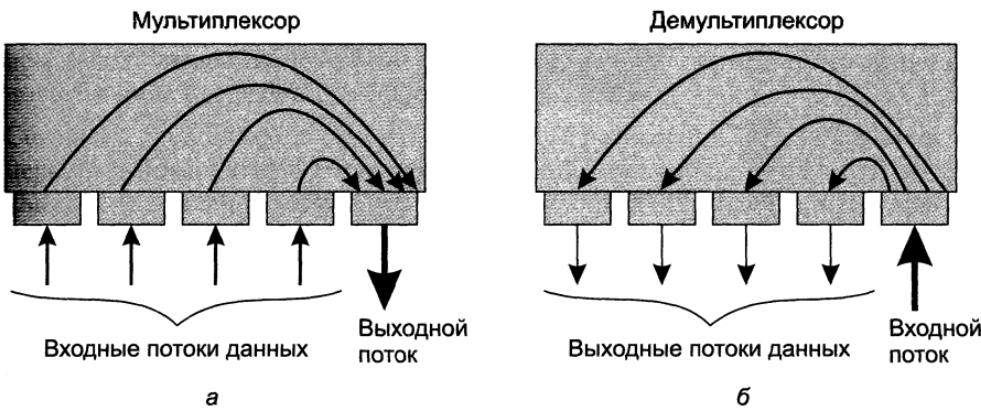


Рис. 1.25. Мультиплексор и демультиплексор

Разделение физической среды

Линия связи может разделяться не только между *потоками*, но и между *интерфейсами*. Во всех приведенных ранее примерах к каждой линии связи подключались только два интерфейса. В компьютерных сетях используется и другой вид подключения, когда к одной линии связи подключается больше,

чем два интерфейса. Именно такой случай показан на рис. 1.26, где потоки, выходящие из узлов 3, 4, 5 и 10, идут по одной линии связи. Такое множественное подключение интерфейсов порождает новую технику использования канала связи, которая получила название **разделение среды**.

Разделяемой средой (shared medium) называется физическая среда передачи данных, к которой непосредственно подключено несколько узлов сети. Причем в каждый момент времени только один из узлов получает доступ к разделяемой среде и задействует ее для передачи данных другому узлу, подключенному к этой же среде.

В качестве разделяемой среды может использоваться коаксиальный кабель, витая пара, оптическое волокно или радиоволны.

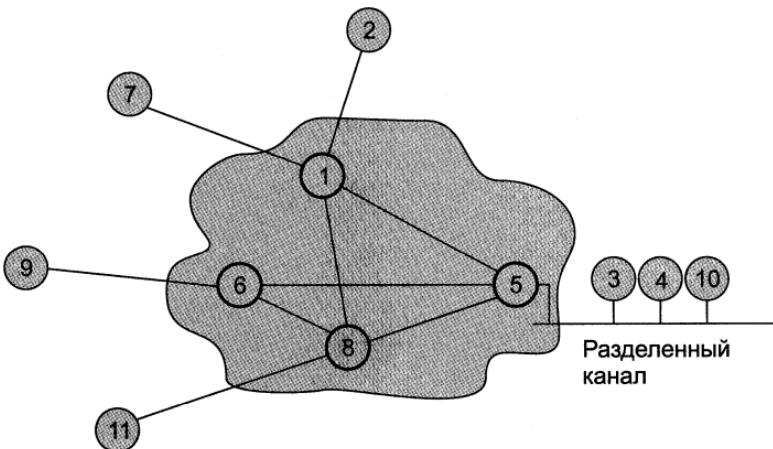


Рис. 1.26. Разделение канала между интерфейсами

На первый взгляд может показаться, что механизм разделения среды очень похож на механизм мультиплексирования потоков — в том и другом случаях по линии связи передаются несколько потоков данных. Однако здесь есть принципиальное различие, и оно заключается в том, как контролируется (управляется) линия связи. При мультиплексировании дуплексная линия связи в каждом направлении находится под полным контролем одного коммутатора, который решает, какие потоки разделяют общий канал связи. Например, коммутатор 8 управляет процессом мультиплексирования потоков в линии связи 8–5 в направлении от 8-го к 5-му, а коммутатор 5 полностью контролирует аналогичный процесс в направлении от 5-го к 8-му.

При разделении среды линия связи находится под управлением нескольких коммутаторов, которым приходится согласованно решать, каким образом использовать физическую среду этой линии связи совместно.

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**, другие — **децентрализованный**. Если мы обратимся к организации работы компьютера, то увидим, что доступ к системной шине компьютера, которую совместно используют внутренние блоки компьютера, управляется централизовано — либо процессором, либо специальным арбитром шины.

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

Для локальных сетей разделяемая среда до сравнительно недавнего времени была основным механизмом использования каналов связи, который применялся во всех технологиях локальных сетей — Ethernet, ArcNet, 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. Например, для создания сети Ethernet на коаксиальном кабеле никакого другого сетевого оборудования, кроме сетевых адаптеров компьютеров и самого кабеля, не требуется. Наращивание количества компьютеров в локальной сети Ethernet на коаксиальном кабеле выполнялось также достаточно просто — путем присоединения нового отрезка кабеля к существующему.

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 масштабируемость¹. Низкая производительность объясняется тем, что пропускная способность канала связи делится между всеми компьютерами сети. Например, если локальная сеть Ethernet состоит из 100 компьютеров и использует для их связи коаксиальный кабель и сетевые адAPTERы, 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, от активности других компьютеров. Наверное, к этому моменту читателю уже понятна причина плохой масштабируемости такой сети — чем больше мы до-

¹ Масштабируемостью называют свойство сети допускать наращивание количества узлов и протяженности линий связи в очень широких пределах, не ухудшая при этом производительности.

бавляем 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 среда — радиоэфир — естественным образом соединяет все конечные узлы, находящиеся в зоне распространения сигнала.

Коммутация пакетов и каналов

Мы уже обсудили в предыдущих разделах, что проблема коммутации узлов компьютерной сети может быть представлена в виде нескольких частных задач, таких как определение потоков данных, поиск и фиксация маршрутов в таблицах коммутации, коммутация интерфейсов в пределах отдельных коммутаторов, мультиплексирование и демультиплексирование потоков. И хотя для каждой из этих задач можно найти множество возможных решений, на практике утвердилось два основополагающих подхода к решению комплекса задач коммутации:

- коммутация каналов;
- коммутация пакетов.

Каждый из этих двух подходов имеет свои достоинства и недостатки. Существуют традиционные области применения каждой из техник коммутации, например, телефонные сети строились и продолжают строиться с использованием техники коммутации каналов, а компьютерные сети в подавляющем большинстве основаны на технике коммутации пакетов. Техника коммутации пакетов гораздо моложе своей конкурентки и пытается вытеснить ее из некоторых областей, например, из телефонии (в форме интернет-телефонии, или IP-телефонии), но этот спор пока не решен, и, скорее всего, две техники коммутации будут сосуществовать еще долгое время, дополняя друг друга. Тем не менее, по долгосрочным прогнозам многих специалистов, будущее принадлежит технике коммутации пакетов, как более гибкой и универсальной.

Далее нам понадобятся несколько новых понятий, применимых к обоим типам сетей.

Предложенная нагрузка — это скорость поступления данных от пользователя на вход сети, измеряемая в битах (или в килобитах, мегабитах и т. д.) в секунду.

Скорость передачи данных — это *фактическая* скорость потока данных, прошедшего через сеть. Эта скорость может быть меньше, чем предложенная нагрузка, если сеть по какой-то причине теряет часть данных.

Пропускная способность линии связи представляет собой *максимально возможную* скорость передачи данных для этой линии связи.

Сети с коммутацией каналов

Сети, построенные на принципе коммутации каналов, имеют богатую историю — они ведут начало от первых телефонных сетей, а сегодня нашли широкое применение и в мире телекоммуникаций, являясь основой создания высокоскоростных магистральных каналов связи. Первые сеансы связи между компьютерами были осуществлены через телефонную сеть, то есть так же с применением техники коммутации каналов, а пользователи, которые получают доступ в Интернет по модему, и сегодня обслуживаются этими сетями, так как их данные доходят до оборудования провайдера по местной телефонной 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**¹. Соответственно, глобальным признаком потока является пара адресов (телефонных номеров) абонентов, связывающихся между собой. Для всех возможных потоков заранее определяются маршруты. Маршруты в сетях с коммутацией каналов либо задаются «вручную» администратором сети, либо находятся автоматически с привлечением специальных программных и аппаратных средств. Маршруты фиксируются в таблицах, 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 понятие элементарного канала.

¹ Термин «абонент» принят в телефонии для обозначения конечного узла. Так как все мы — многолетние пользователи телефонной сети, то далее мы будем сопровождать наше объяснение принципа работы сетей с коммутацией каналов примерами из области телефонии.

Элементарный канал (или просто **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.

ПРИМЕЧАНИЕ

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

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 предложенной нагрузки. Например, в традиционных телефонных сетях наиболее распространенным значением элементарного канала является сегодня скорость 64 Кбит/с — это минимально достаточная скорость для качественной цифровой передачи голоса¹.

Линии связи в сетях с коммутацией пакетов (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, соединяющие коммутаторы, — 480 или 1920 каналов.

Обратимся к фрагменту сети, изображенному на рис. 1.27. Предположим, что эта сеть характеризуется элементарным каналом P бит/с. В сети существуют линии связи разной пропускной способности, состоящие из 2, 3, 4 и 5

¹ Вы можете прочитать про преобразование голоса в цифровую форму в разделе «Технология ATM» главы 4.

элементарных 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: *S₁*, *S₂*, *S₃* и *S₄*. Предположим также, что интенсивность информационного потока между абонентами не превосходит 2Р 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 эти элементарные каналы, необходимые абонентам *A* и *B*, обозначены толстыми линиями.

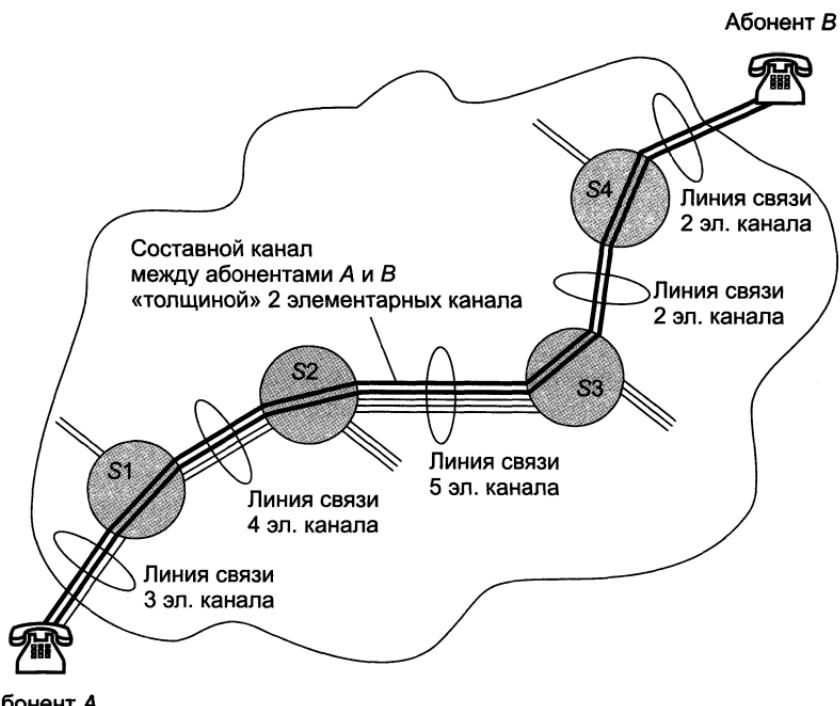


Рис. 1.27. Составной канал в сети с коммутацией каналов

Составной канал

Связь, построенная путем коммутации (соединения) элементарных каналов, называют **составным каналом**.

В рассматриваемом примере для соединения абонентов *A* и *B* был создан составной канал «толщиной» в 2 элементарных канала. Если изменить наше предположение и считать, что предложенная нагрузка гарантированно

не превысит P бит/с, то абонентам будет достаточно иметь в своем распоряжении составной канал, «толщиной» в 1 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;
- ❑ 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 посыпать 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, 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 (см. рис. 1.27). 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.

Для того 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), посыпает в коммутационную сеть **запрос** — сообщение, в котором содержится адрес вызываемого абонента, например абонента *B*¹.

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

Запрос перемещается по *маршруту*, определенному для информационного потока данной пары абонентов. При этом используются глобальные таблицы коммутации, ставящие в соответствие *глобальному* признаку потока (адресу вызываемого абонента) идентификатор выходного интерфейса коммутатора (как уже упоминалось, такие таблицы часто называют также таблицами маршрутизации).

Если в результате прохождения запроса от абонента *A* к абоненту *B* выяснилось, что ничто не препятствует установлению соединения, происходит **фиксация** составного канала. Для этого во всех коммутаторах 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. Только после этого составной канал считается установленным и 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 организует 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 — номеров 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 оба необходимых ему

¹ В телефонной сети посылке запроса соответствует набор телефонного номера.

элементарных канала, составляющих линию связи коммутаторов $S3$ и $S4$, уже выделены соединению абонентов A и B (рис. 1.28). При отказе в установлении соединения сеть информирует вызывающего абонента специальным сообщением¹. 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.

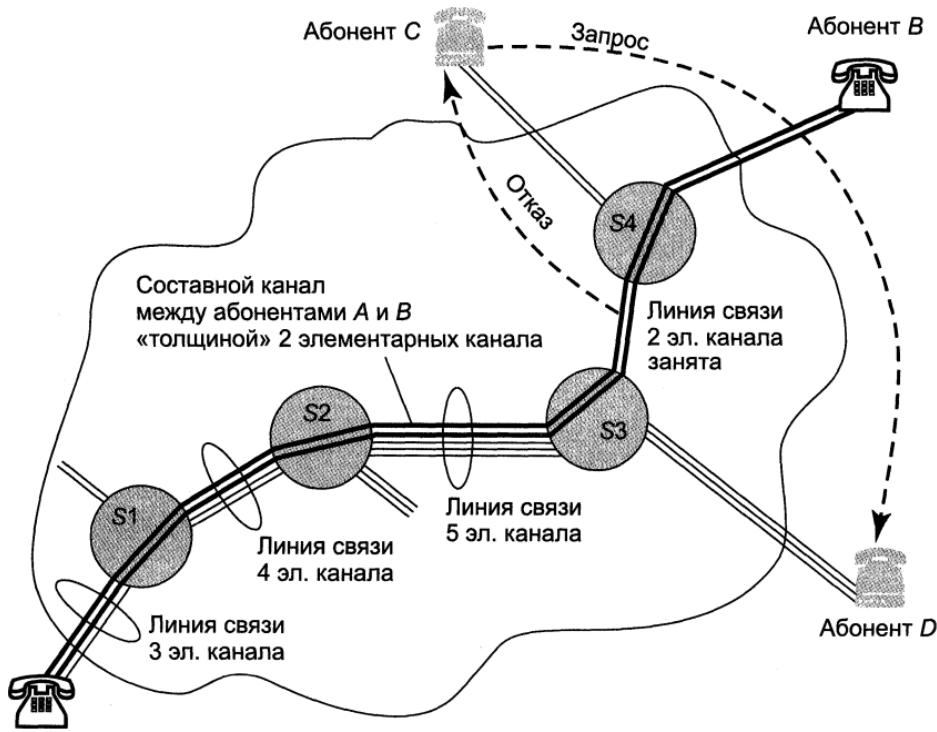


Рис. 1.28. Отказ в установлении соединения в сети с коммутацией каналов

Мы описали процедуру установления соединения в *автоматическом динамическом режиме*, основанном на способности абонентов отправлять в сеть служебные сообщения — запросы на установление соединения и способности узлов сети обрабатывать такие сообщения. Подобный режим используется телефонными сетями: телефонный аппарат генерирует запрос, посыпая в сеть импульсы (или тоновые сигналы), кодирующие номер вызываемого абонента, а сеть либо устанавливает соединение, либо сообщает об отказе сигналами «занято».

¹ Телефонная сеть в этом случае передает короткие гудки — сигнал «занято». Некоторые телефонные сети различают события «сеть занята» и «абонент занят», передавая гудки с разной частотой или используя разные тона.

Однако это не единственный возможный режим работы сети с коммутацией каналов, существует и другой *статический ручной режим* установления соединения. Этот режим характерен для случаев, когда необходимо установить составной канал не на время одного сеанса связи абонентов, а на более долгий срок. Создание такого **долговременного канала** не могут инициировать абоненты, он создается **администратором** сети. Очевидно, что статический ручной 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 всяких возможных неопределенностей значительная часть работы по организации информационного обмена выполняется заранее, 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 одну непрерывную «трубу» (составной канал), имеющую «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. 1.29).

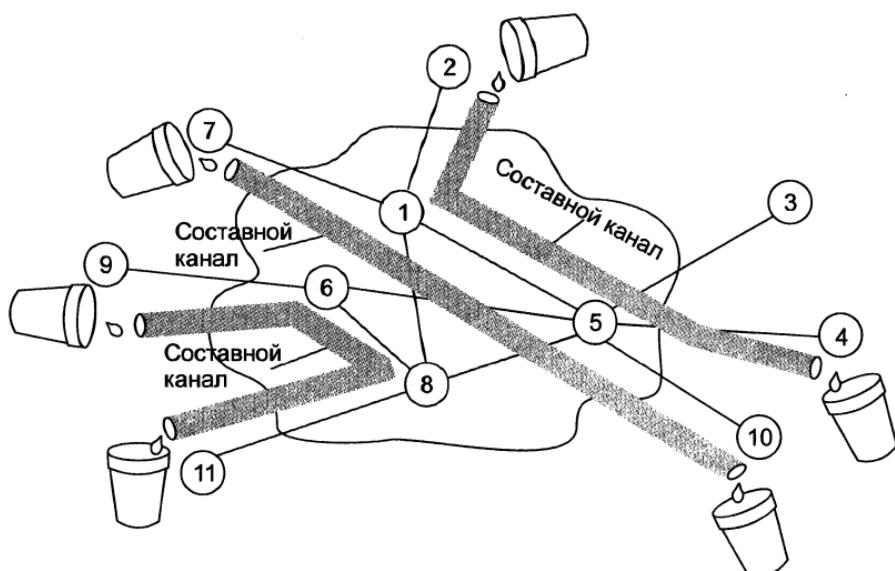


Рис. 1.29. Сеть с коммутацией каналов как система трубопроводов

Неэффективность при передаче 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 с телефонными разговорами это снижение оказывается вполне приемлемым и позволяет широко использовать сети с коммутацией каналов для передачи голосового трафика.

Гораздо больше снижает эффективность сети с коммутацией каналов передача так называемого *компьютерного трафика* – то есть трафика, генерируемого приложениями, с которыми 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 производительности 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, другими словами, «перевозит воздух».

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

Сети с коммутацией пакетов

Сети, основой которых является коммутация пакетов, сравнительно молоды, они появились в конце 60-х годов как результат экспериментов с первыми глобальными компьютерными сетями.

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, выделенных исключительно для них каналов связи. 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.

Пакет снабжен заголовком, в котором содержится адрес назначения и другая вспомогательная информация (длина поля данных, контрольная сумма и др.), используемая для доставки пакета адресату.

В зависимости от конкретной реализации технологии коммутации пакетов пакеты могут иметь фиксированную или переменную длину, может меняться также и 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).

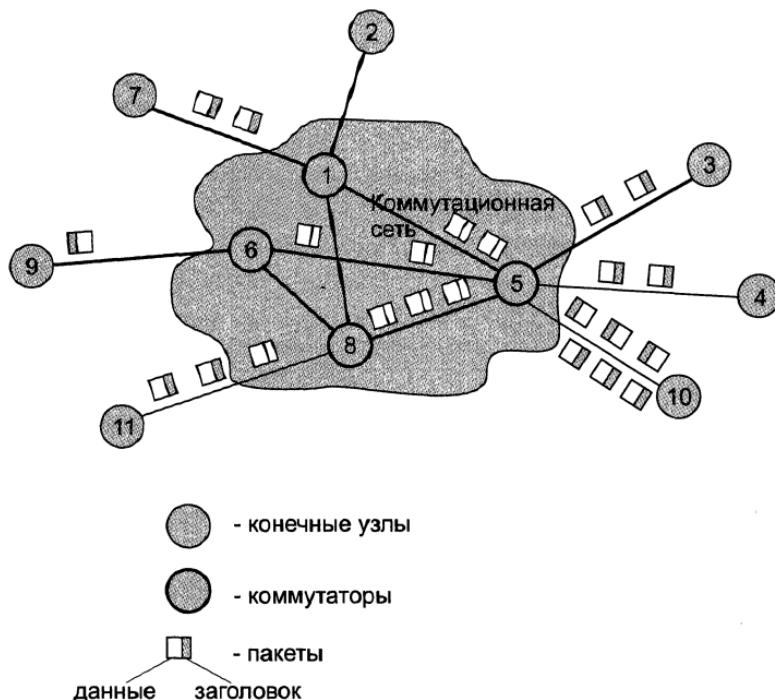


Рис. 1.30. Передача данных по сети в виде пакетов

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

В сетях с коммутацией пакетов информационным потоком называют совокупность пакетов, объединенных набором общих признаков, выделяющих их из общего сетевого трафика. Чаще всего в качестве признака потока выступает адрес назначения. Таким образом, все пакеты, направляемые по одному адресу, образуют поток.

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. 1.30).

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 встретиться 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 приходить к месту назначения не в том порядке, в котором они были отправлены.

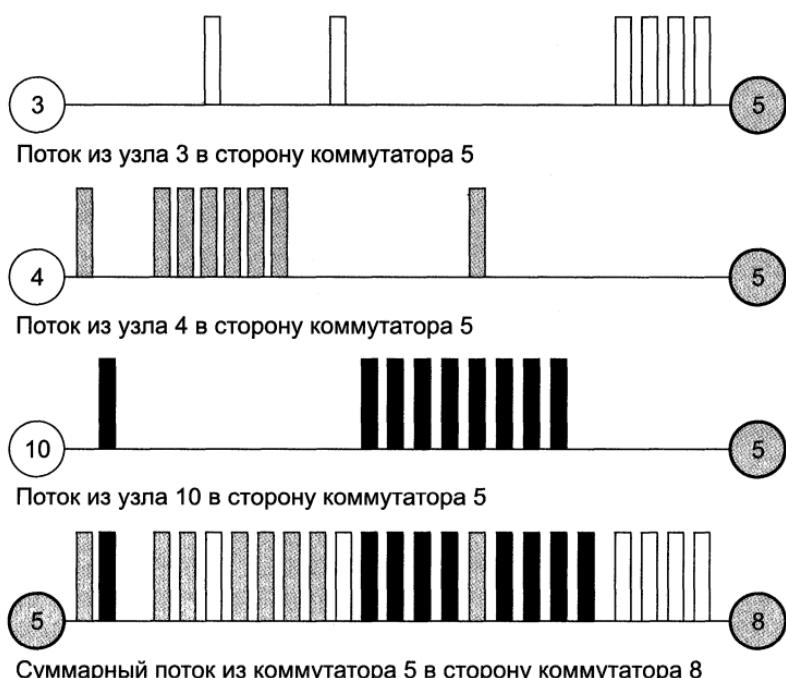


Рис. 1.31. Сглаживание трафика в сетях с коммутацией пакетов

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 ее пропускная способность используется более рационально, без длительных простоев. Этот эффект иллюстрируется рис. 1.31, на котором показаны неравномерные потоки пакетов, поступающие от конечных узлов 3, 4 и 10 в сети, изображенной на рис. 1.30. 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.

Буферизация пакетов

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

Главное отличие пакетных коммутаторов¹ от коммутаторов в сетях с коммутацией каналов состоит в том, что они имеют внутреннюю **буферную память** для временного хранения пакетов.

Коммутатору нужны буфера для согласования скоростей передачи данных в линиях связи, подключенных к его интерфейсам. Действительно, если скорость поступления 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. 1.32).

Буферизация необходима пакетному коммутатору также 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.

Поскольку объем буферов в коммутаторах ограничен, иногда происходит потеря пакетов из-за переполнения буферов при временной перегрузке части сети, когда совпадают периоды пульсации нескольких информационных потоков. Для сетей с коммутацией пакетов потеря пакетов является обычным явлением, и для компенсации таких потерь в данной сетевой технологии предусмотрен ряд специальных механизмов, которые мы рассмотрим позже.

¹ Для простоты будем далее называть коммутаторы сетей с коммутацией пакетов «пакетными коммутаторами».

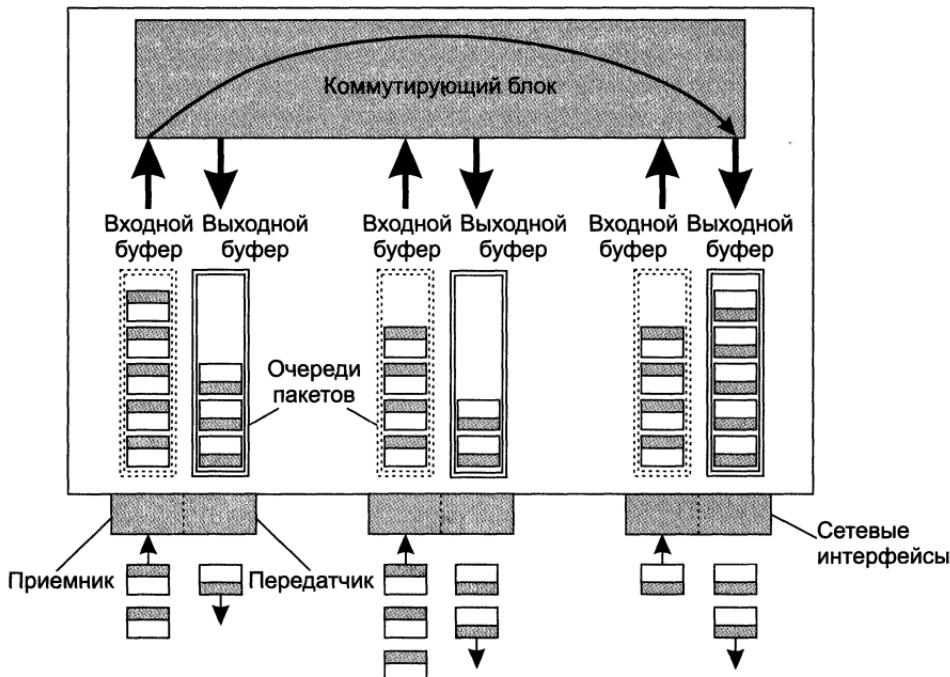


Рис. 1.32. Буферы и очереди пакетов в коммутаторе

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.

Процедура обработки пакета определяется только значениями параметров, которые он несет в себе, и текущим состоянием сети (например, в зависимости от ее нагрузки пакет может стоять в очереди на обслуживание большее или меньшее время). Однако 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 коммутации¹, 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. 1.33 показан пакет, отправленный из конечного узла 7 в адрес конечного узла 4. При поступлении этого пакета в коммутатор 1 в таблице коммутации определяется направление продвижения этого пакета, а именно, он должен быть передан на интерфейс *B*, который ведет его к следующему по маршруту коммутатору 5. На коммутаторе 5 будет проделана подобная процедура, в результате которой пакет попадет в узел назначения 4.

Таблица коммутации коммутатора 1

Признак (адрес)	Интерфейс
2	A
3	B
4	B
5	B
6	C
7	D
8	C
9	C
10	B
11	C

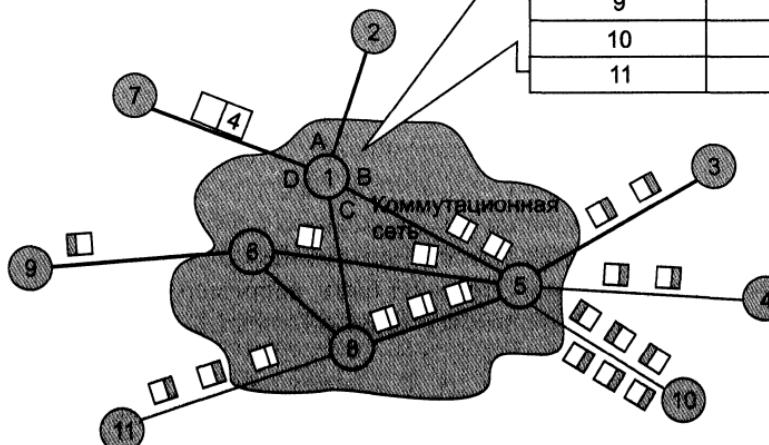


Рис. 1.33. Дейтаграммный метод продвижения пакетов

¹ Напомним, что в разных технологиях для обозначения таблиц, имеющих указанное выше функциональное назначение, могут использоваться другие термины (таблица маршрутизации, таблица продвижения и др.).

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. Этот метод не гарантирует доставку пакета, доставка происходит по возможности, или, как говорят, **с максимальными усилиями (best effort)**.

Передача с установлением логического соединения

Следующий рассматриваемый нами способ продвижения пакетов основывается на знании «истории» обмена пакетами и позволяет более рационально по сравнению с дейтаграммным способом обрабатывать пакеты. Например, за счет установления соединения может быть обеспечена повышенная надежность передачи данных путем нумерации пакетов, отбрасывания дубликатов, упорядочивания поступивших и повторения передачи потерянных пакетов.

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**.

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).

Процедура установления соединения состоит обычно из трех шагов.

1. Узел-инициатор соединения отправляет узлу-получателю *служебный пакет* с предложением установить соединение.
2. Если узел-получатель согласен с этим, то он посыпает в ответ другой служебный пакет, подтверждающий установление соединения и предлагающий некоторые параметры, которые будут использоваться в рамках данного логического соединения. Это могут быть, например, идентификатор соединения, количество кадров, которые можно отправить без получения подтверждения, и т. п.
3. Узел-инициатор соединения может закончить процесс установления соединения отправкой третьего служебного пакета, в котором сообщит, что предложенные параметры ему подходят.

После того как соединение установлено и все параметры согласованы, конечные узлы начинают передачу собственно данных. Пакеты данных обрабатываются коммутаторами точно так же, как и при дейтаграммной передаче: из заголовков пакетов извлекаются адреса назначения и сравниваются

с записями в таблицах коммутации, содержащих информацию о следующих шагах по маршруту. Так же как 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-правителем, а расшифровка — узлом-получателем. Аналогично, для обеспечения в рамках логического соединения надежности всю работу по нумерации пакетов, отслеживанию номеров доставленных и недоставленных пакетов, посылке копий и отбрасыванию дубликатов берут на себя конечные узлы.

ПРИМЕЧАНИЕ

Некоторые параметры логического соединения могут рассматриваться так же, как признаки информационного потока между узлами, установившими это логическое соединение.

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 данные передаются 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 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 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**.

Виртуальные каналы прокладываются для *устойчивых* информационных потоков. С целью выделения потока данных из общего трафика каждый пакет этого потока помечается специальным видом признака — **меткой**.

Так же как в сетях с установлением логических соединений, прокладка виртуального канала начинается с отправки из узла-источника 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 каждом 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 меткой¹.

После прокладки виртуального канала сеть может передавать по нему соответствующий поток данных. Во всех пакетах, которые переносят пользовательские данные, адрес назначения уже не указывается, его роль играет 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. 1.34 показана сеть, в которой проложены два виртуальных канала (Virtual Channel, VC), идентифицируемые метками VC1 и VC2. Первый проходит от конечного узла с адресом 7 до конечного узла с адресом 4 через промежуточные коммутаторы 1 и 5. Второй виртуальный канал, VC2, обеспечивает продвижение данных по пути 2—1—8—11. В общем случае между двумя конечными узлами может быть проложено несколько виртуальных каналов, например, еще один виртуальный канал между узлами 7 и 4 мог бы проходить через промежуточный коммутатор 8. На рисунке показаны два пакета, несущие в своих заголовках метки потоков VC1 и VC2, которые играют роль адресов назначения.

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 и таблицы коммутации в этом случае намного короче, а следовательно, анализ такой таблицы занимает у коммутатора меньше времени. По этой же причине метка короче адреса конечного узла и заголовок пакета в сетях с виртуальными каналами переносит по сети вместо длинного адреса компактный идентификатор потока.

¹ Эта метка в различных технологиях называется по-разному: номер логического канала (Logical Channel Number, LCN) в технологии X.25, идентификатор соединения уровня канала данных (Data Link Connection Identifier, DLCI) в технологии Frame Relay, идентификатор виртуального канала (Virtual Channel Identifier, VCI) в технологии ATM.

Таблица коммутации
коммутатора 1

Адрес назначения	Интерфейс
VC 1	B
VC 2	C

Виртуальный канал VC 1 7-1-5-4

Виртуальный канал VC 2 2-1-8-11

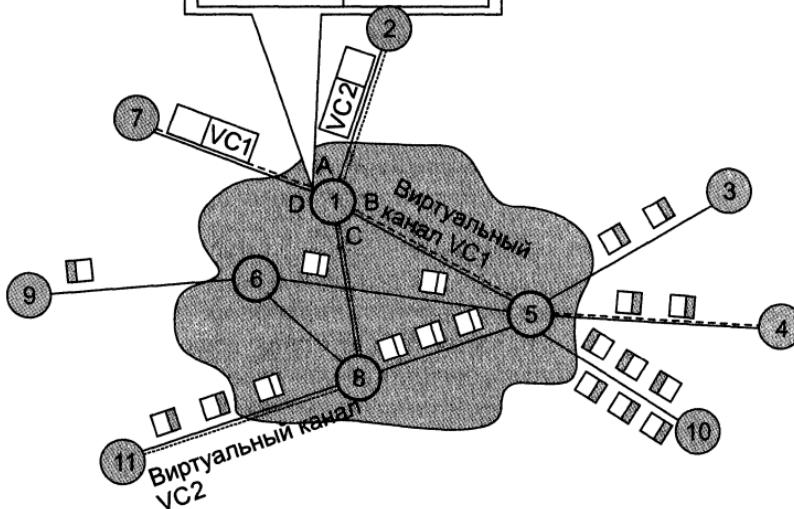


Рис. 1.34. Продвижение пакетов на основе виртуальных каналов

ПРИМЕЧАНИЕ

Использование в сетях техники виртуальных каналов не делает их сетями с коммутацией каналов. Хотя в подобных сетях применяется процедура предварительного 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, дейтаграммный протокол IP используется для передачи данных между отдельными сетями, составляющими Интернет. 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 ATM и Frame Relay, поддерживающих виртуальные каналы.

препятствий друг другу. Очевидно, что для колонны автомобилей создаются наиболее благоприятные условия для движения, но при этом автомобили теряют свою самостоятельность, превращаясь в поток, из которого нельзя «свернуть» в сторону. Дорога при такой организации движения используется нерационально, так как полоса простирает значительную часть времени, как и полоса пропускания в сетях с коммутацией каналов.

В табл. 1.1 сведены свойства обоих видов сетей. На основании этих данных можно аргументированно утверждать, в каких случаях рациональнее использовать сети с коммутацией каналов, а в каких — с коммутацией пакетов.

Таблица 1.1. Сравнение сетей с коммутацией каналов и пакетов

Коммутация каналов	Коммутация пакетов
Необходимо предварительно устанавливать соединение	Отсутствует этап установления соединения (дейтаграммный способ)
Адрес требуется только на этапе установления соединения	Адрес и другая служебная информация передается с каждым пакетом
Сеть может отказать абоненту в установлении соединения	Сеть всегда готова принять данные от абонента
Гарантированная пропускная способность для взаимодействующих абонентов	Пропускная способность сети для абонентов неизвестна, задержки передачи носят случайный характер
Трафик реального времени передается без задержек	Ресурсы сети используются эффективно при передаче пульсирующего трафика
Высокая надежность передачи	Возможные потери данных из-за переполнения буферов
Нерациональное использование пропускной способности каналов, снижающее общую эффективность сети	Автоматическое динамическое распределение пропускной способности физических каналов в соответствии с фактической интенсивностью трафика абонентов

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 ему свой трафик, причем в обеих сетях пропускная способность линий связи одинакова. Очевидно, что более эффективной с точки зрения временных затрат для нашего пользователя была бы работа в сети с коммутацией каналов, где ему в единичное владение предоставляется зарезервированный канал связи. При этом способе все данные поступали бы адресату без задержки. Тот факт, что значительную часть времени зарезервированный канал будет простираивать (во время пауз), нашего пользователя не волнует — ему важно быстро решить собственную задачу.

Сравнение сетей с коммутацией пакетов и каналов

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

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. Слишком интенсивный трафик, не соответствующий пропускной способности дороги, приводит к перегруженности дорог, в результате автомобили стоят в пробках, что соответствует очередям пакетов в коммутаторах.

На перекрестках происходит «коммутация» потоков автомобилей, каждый из автомобилей выбирает подходящее направление, 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 светофор определяет очередность пересечения перекрестка потоками автомобилей. Еще больше похоже на работу коммутатора поведение регулировщика трафика, который может выбрать для продвижения не только поток автомобилей в целом, но и отдельный автомобиль.

Как и в сетях с коммутацией пакетов, к образованию заторов на дорогах приводит неравномерность движения автомобилей. Так, даже кратковременное снижение скорости одного автомобиля на узкой дороге может создать большую пробку, которой не было бы, если бы все автомобили всегда двигались с одной и той же скоростью и равными интервалами.

А теперь попробуем найти общее в автомобильном движении и в сетях 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. Для того чтобы колонна двигалась без препятствий, заранее для ее движения 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 еще за некоторое время до начала движения колонны, и это резервирование отменяется 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

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. Кроме того, дополнительное время тратится на формирование пакетов и передачу информации, содержащейся в заголовках пакетов. В то же время некоторое снижение производительности по отношению к отдельным пользователям компенсируется более эффективной обработкой сетевого трафика в целом, равномерной загрузкой линий связи, отсутствием простоев оборудования.

Типы компьютерных сетей

Классифицировать компьютерные сети можно, основываясь на самых разных признаках: выбранной технологии, территориальной протяженности, области применения, характере предоставляемых услуг, преимущественно используемом оборудовании и программном обеспечении и т. д. В этом разделе приведены описания нескольких основных типов компьютерных сетей.

Глобальные сети

Одним из наиболее очевидных способов классификации компьютерных сетей является разделение их по территориальному признаку на локальные и глобальные.

Глобальные сети (Wide Area Network, WAN) объединяют компьютеры, находящиеся на больших расстояниях друг от друга: в различных городах, в разных странах и на разных континентах.

Глобальные сети появились раньше, чем локальные, — это произошло во второй половине 60-х, когда царили большие компьютеры, так называемые мэйнфреймы. Крупное предприятие обычно имело тогда в своих филиалах, расположенных в разных городах, по одному компьютеру. Объединение этих компьютеров между собой давало предприятию явные преимущества, обеспечивая, например, доступ всех сотрудников предприятия, независимо от их местоположения, к одной и той же центральной базе данных. Потребность предприятий в создании локальных сетей в то время еще не созрела — в одном здании просто нечего было объединять в сеть, так как из-за высокой стоимости компьютеров предприятия не могли себе позволить роскошь приобретения большого количества компьютеров.

Создание глобальных компьютерных сетей началось с решения более простой задачи — доступа к компьютеру с терминалов, удаленных от него на многие сотни, а то и тысячи километров. Терминалы соединялись с компьютерами через телефонные сети с помощью модемов. Такие сети позволяли

многочисленным пользователям получать удаленный доступ к разделяемым ресурсам нескольких мощных компьютеров. Затем появились системы, в которых наряду с удаленными соединениями типа *терминал–компьютер* были реализованы и удаленные связи типа *компьютер–компьютер*.

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 сетевыми службами.

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 популярной ныне глобальной сети – **Интернет**.

При построении глобальных сетей были впервые предложены и отработаны многие основные идеи, лежащие в основе современных компьютерных сетей. Такие, например, как концепция коммутации пакетов и многоуровневое построение коммуникационных протоколов.

Так как прокладка высококачественных линий связи на большие расстояния обходится очень дорого, в первых глобальных сетях часто использовались уже существующие каналы связи, изначально предназначенные совсем для других целей. Например, в течение многих лет глобальные сети строились на основе аналоговых телефонных каналов. Поскольку скорость передачи дискретных компьютерных данных по таким каналам была очень низкой (десятки килобитов в секунду), набор предоставляемых услуг в глобальных сетях такого 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 глобальные сети, построенные с использованием каналов связи низкого качества, отличались сложными процедурами контроля и восстановления данных.

Прогресс глобальных компьютерных сетей во многом определялся прогрессом в области каналов связи. Появление качественных и скоростных каналов привело к эволюции услуг, предоставляемых глобальными сетями. Например, веб-сервис изначально был разработан для «медленных» каналов, которые преобладали в начале 90-х, поэтому главным элементом веб-страниц был текст и простая графика. Сегодня веб-дизайнеры для создания эффектных и эффективных страниц используют все типы мультимедия – звук, видео и качественную графику.

Глобальные сети могут объединять как отдельные компьютеры, так и локальные сети. Подключение крупного компьютера непосредственно к ком-

мутатору глобальной сети было характерно для ранней эпохи развития компьютерных сетей, но сегодня такой способ нерационален, так как даже суперкомпьютеры объединяются локальной сетью с другими компьютерами.

Топология связей глобальной сети обычно не сводится к какой-нибудь регулярной структуре, такой как, например, звезда или кольцо, хотя такие регулярные структуры и могут входить в общую топологию сети как элементы (рис. 1.35). При создании глобальной сети на ее топологию влияет много факторов, в первую очередь географическое расположение узлов сети, доступность каналов связи между ними и оценки уровней трафика, который будет передаваться между узлами сети. Последний фактор важен из-за того, что каналы дальней связи дороги и увеличение их пропускной способности — дело непростое, не сводящееся к подключению к свободным портам коммутатора еще одного кабеля, как в случае локальной сети. Поэтому при создании глобальной сети обычно стараются выбрать пропускную способность каналов такой, чтобы ее хватало с прицелом на рост трафика в течение нескольких лет.

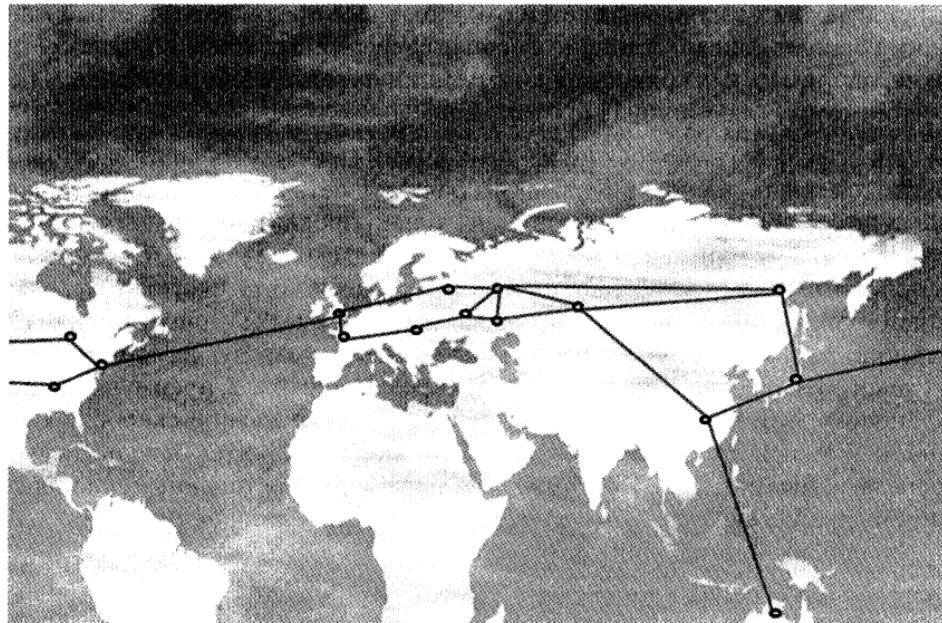


Рис. 1.35. Пример топологии глобальной сети

Несмотря на выросшее качество дальних каналов связи отказы их возможны, а цена отказа канала может быть очень велика, так как по каналу передаются данные большого количества абонентов. Поэтому в глобальных сетях обычно стремятся к топологии, обеспечивающей резервирование маршрутов трафика, то есть наличие более чем одного пути между узлами сети, особенно между теми узлами, через которые проходит значительная часть трафика.

Локальные сети

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. В общем случае локальная сеть представляет собой коммуникационную систему, принадлежащую одной организации.

Локальные сети появились в 70-х годах как средство объединения **мини-компьютеров**, которые к тому времени благодаря сравнительно невысокой стоимости и хорошим функциональным возможностям стали реальными конкурентами мэйнфреймов. Мини-компьютеры решали задачи управления технологическим оборудованием, складом и другие задачи уровня отдела предприятия. Таким образом, появилась концепция распределения компьютерных ресурсов по всему предприятию. Очень скоро у пользователей возникла потребность в автоматическом режиме обмена компьютерными данными с пользователями других подразделений. Ответом на эту потребность и явилось появление первых локальных вычислительных сетей.

Мощным стимулом для дальнейшего развития технологий локальных сетей стало появление **персональных компьютеров**. 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 мини-компьютеры и мэйнфреймы.

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 построение локальной сети из сложной технической задачи в рутинную работу. Для создания сети достаточно было приобрести стандартный кабель, сетевые адAPTERы и коммутаторы соответствующего стандарта, например Ethernet, вставить адAPTERы в компьютеры, присоединить их стандартными кабелями со стандартными разъемами к коммутаторам и установить на компьютеры одну из популярных операционных систем, поддерживающих сетевые операции, например, Microsoft Windows или UNIX. Топологии связей локальных сетей также стандартные: звезда, кольцо или же показанное в качестве примера на рис. 1.36 дерево.

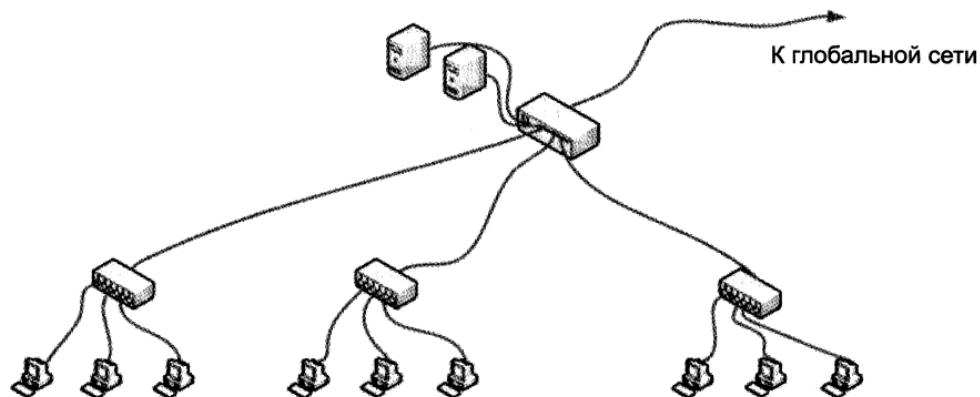


Рис. 1.36. Пример локальной сети с древовидной топологией

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. Действительно, в отличие от глобальной в локальной сети, пользователь освобождается от запоминания сложных идентификаторов разделяемых ресурсов. Для этих целей система предоставляет ему список ресурсов в удобной для восприятия форме, например в виде древовидной иерархической структуры («дерева» ресурсов). Еще один прием, рационализирующий работу пользователя в локальной сети, состоит в том, что после соединения с удаленным ресурсом пользователь получает возможность обращаться к нему с помощью тех же команд, что и для работы с локальными ресурсами. Последствием и одновременно движущей силой такого прогресса стало появление огромного числа непрофессиональных пользователей, освобожденных от необходимости изучать специальные (и достаточно сложные) команды для работы в сети.

Конец 90-х выявил явного лидера среди технологий локальных сетей — семейство **Ethernet**. Простые алгоритмы работы предопределили 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. Важно также, что все технологии Ethernet очень близки друг другу по принципам функционирования, что упрощает обслуживание и интеграцию этих сетей.

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

В последние годы различия между локальными и глобальными типами сетевых технологий стали сглаживаться. Изолированные ранее локальные сети начали объединять друг с другом, при этом в качестве связующей среды использовались глобальные сети. Тесная интеграция локальных и глобальных сетей привела к значительному взаимопроникновению соответствующих технологий.

Эти сети 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.

Составные сети

До этого момента мы по умолчанию предполагали, что все конечные узлы (компьютеры, телефоны, датчики, исполнительные устройства) подключаются к сети одного типа, построенной на основе единой технологии. Однако отнюдь не редкими являются случаи, когда пользователи взаимодействуют через среду, образованную несколькими связанными между собой сетями разного типа. Такие сети-компоненты мы будем называть **подсетями**, хотя этот термин имеет и другой смысл, с которым мы познакомимся позже. Технологии, на основе которых построены подсети, могут отличаться друг от друга, например, системой адресации, форматом используемых кадров и алгоритмами коммутации. Так, в одной подсети может применяться дейтаграммный способ продвижения данных, а в другой — техника предварительно устанавливаемых виртуальных каналов.

Сеть, образованная путем соединения нескольких подсетей разного типа, называется **составной сетью, или интернетом**¹.

На рис. 1.37 показан пример составной сети. Здесь несколько локальных сетей Ethernet, принадлежащих одному предприятию, соединены между собой через глобальную сеть провайдера, построенную на основе технологии ATM. Технология локальных сетей Ethernet не может решить задачу связывания конечных узлов, принадлежащих разным сетям. Для решения этой проблемы необходимо привлечение специально предназначенных дополнительных сетевых средств.

Технология, позволяющая соединять в единую сеть множество сетей, в общем случае построенных на основе разных технологий, называется технологией **межсетевого взаимодействия (internetworking)**.

¹ Не следует путать интернет (со строчной буквы) с Интернетом (с прописной буквы). Интернет — это самая известная и охватывающая весь мир реализация составной сети, построенная на основе технологии TCP/IP.

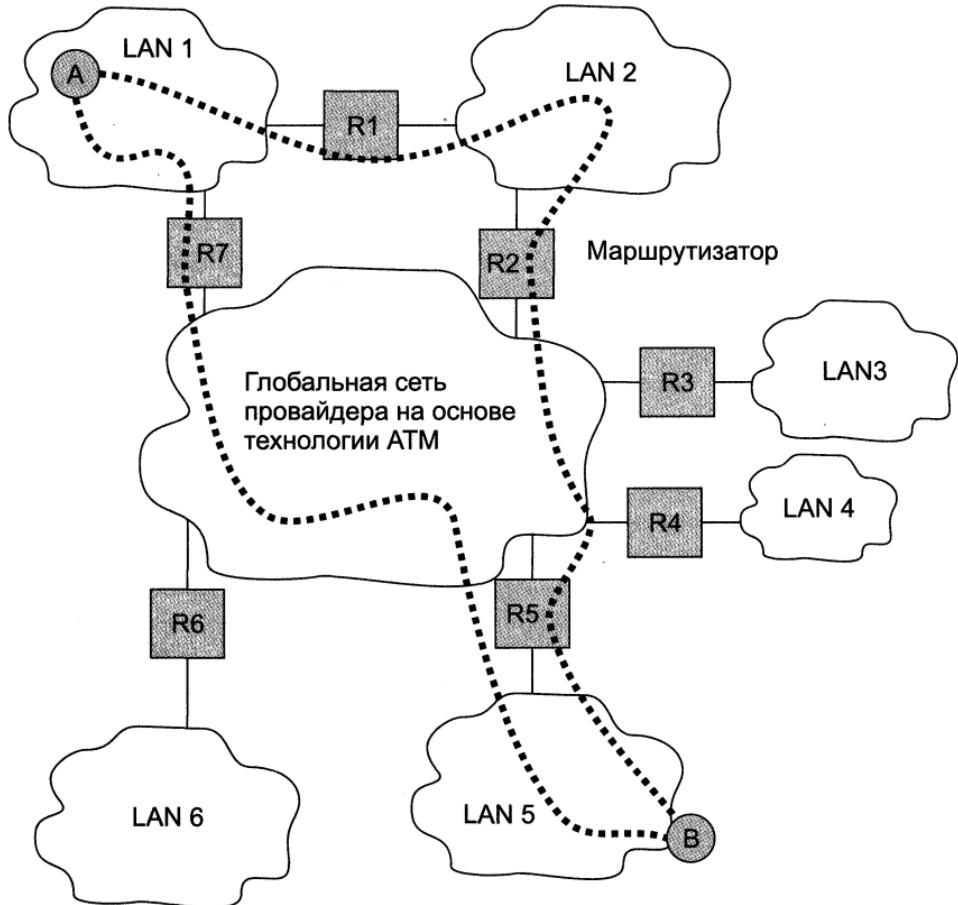


Рис. 1.37. Составная сеть

Технология межсетевого взаимодействия предусматривает использование между соединяемыми сетями специального пограничного устройства, называемого **маршрутизатором**.

Маршрутизатор является частным видом коммутатора (в том широком смысле, которым мы наделяли этот термин выше), который коммутирует информационные потоки между сетями. 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. Причем интерфейс маршрутизатора должен поддерживать ту технологию, на основе которой построена подключаемая к этому интерфейсу сеть. Таким образом, все интерфейсы маршрутизатора можно считать узлами разных сетей.

Маршрутизатор может быть реализован программно, на базе универсального компьютера (например, типовая конфигурация UNIX или Windows включает программный модуль маршрутизатора). Однако чаще маршрутизаторы реализуются на базе специализированных аппаратных платформ, на которых установлено соответствующее программное обеспечение.

Итак, чтобы связать сети, показанные на рис. 1.37, необходимо соединить все эти сети маршрутизаторами, а также установить дополнительные программные модули, предназначенные для решения задачи межсетевого взаимодействия, на все конечные узлы, пользователи которых желают связываться через составную сеть.

Для того чтобы узлы могли обмениваться данными в пределах всей составной сети, эти узлы должны иметь адреса, уникальные в пределах всей составной сети. Такие адреса называются **сетевыми**, или **глобальными**. Каждый узел составной сети, который намерен обмениваться данными с другими узлами составной сети, должен иметь сетевой адрес наряду с локальным адресом, назначенным ему в его собственной сети.

В пакете, передаваемом через составную сеть, в качестве адреса назначения должен быть указан сетевой адрес, на основании которого определяется маршрут пакета. Маршрут в составной сети описывается последовательностью сетей (или маршрутизаторов), через которые должен пройти пакет, чтобы попасть к адресату. Например, на рис. 1.37 штриховой линией показано два маршрута, по которым могут быть переданы данные от компьютера A к компьютеру B. Маршрутизатор собирает информацию о топологии связей между сетями и на ее основании строит таблицы коммутации, которые в данном случае носят специальное название **таблица маршрутизации**. Задачу выбора маршрута мы уже коротко обсуждали в разделе «Определение маршрутов».

Путь пакета через составную сеть разбивается на участки от одного маршрутизатора до другого, причем каждый участок соответствует пути через отдельную сеть. Для того чтобы передать пакет между соседними маршрутизаторами, используются сетевые средства доставки промежуточной сети. При этом маршрутизатор- отправитель должен «упаковать» передаваемый пакет в поле данных пакета, имеющего формат, принятый в данной промежуточной сети. В качестве адреса назначения в заголовке пакета указывается не сетевой, а локальный адрес интерфейса следующего маршрутизатора. При выполнении этих двух условий доставка пакета между двумя маршрутизаторами может быть осуществлена средствами промежуточной сети. Когда пакет доставлен пограничному маршрутизатору, он «распаковывает» его, извлекает исходный пакет, а затем выполняет те же подготовительные действия, что и предыдущий маршрутизатор, чтобы воспользоваться транспортными возможностями следующей промежуточной сети для доставки пакета следующему маршрутизатору.

Пример

Можно найти аналогию между функционированием составной сети и международной почтовой службы, такой, например, как DHL (рис. 1.38). Представим, что некоторый груз необходимо доставить из города Абра в город Кадабра, причем эти города расположены на разных континентах. Для доставки груза международная почта использует услуги различных региональных перевозчиков: железнодорожный транспорт, морской транспорт, авиаперевозчиков, автомобильный транспорт.

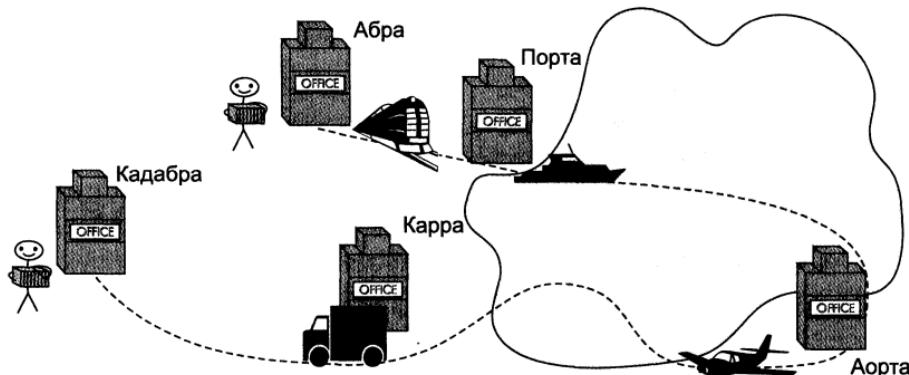


Рис. 1.38. Работа международной почтовой службы

Эти перевозчики могут рассматриваться как аналоги сетей-компонентов, причем каждая «сеть» здесь построена на основе собственной технологии. Из этих региональных перевозчиков международная почтовая компания должна организовать единую слаженно работающую службу (составную сеть). Для этого международная почтовая компания должна, во-первых, продумать маршрут перемещения почты, во-вторых, скоординировать работу в пунктах смены перевозчиков (например, выгружать почту из вагонов и размещать ее в транспортном отсеке самолета). Каждый же перевозчик ответственен только за перемещение почты по своей части пути и не несет никакой ответственности за состояние почты за его пределами.

Телекоммуникационные сети

Телекоммуникационные сети – это обобщенное название для телефонных, телевизионных, радио и глобальных компьютерных сетей. Несмотря на множество очевидных и менее очевидных различий, в их структуре можно найти много общего, и прежде всего схожую архитектуру. Так, в состав любой телекоммуникационной сети входят (рис. 1.39):

- ❑ сети доступа;
- ❑ терминальное оборудование пользователей (возможно, объединенное в сеть);

- магистральная сеть;
- информационные центры.

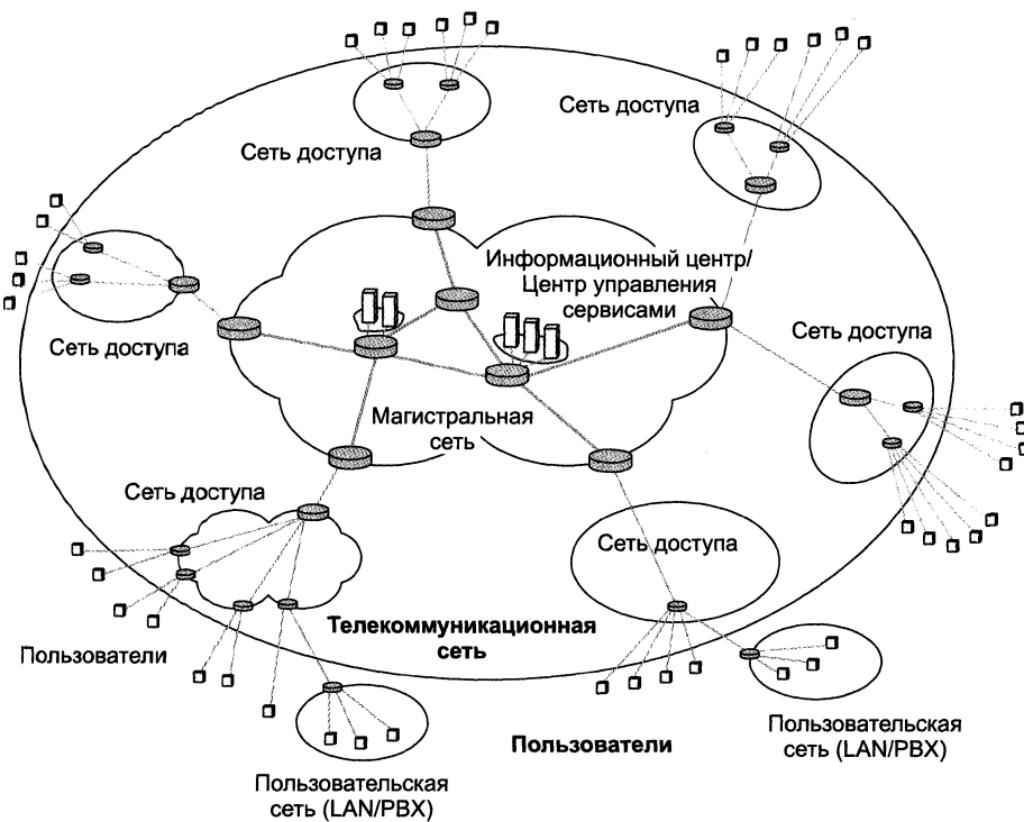


Рис. 1.39. Обобщенная структура телекоммуникационной сети

Как сеть доступа, так и магистральная сеть строятся на основе коммутаторов. Каждый коммутатор оснащен некоторым количеством портов, которые соединяются с портами других коммутаторов.

Сеть доступа составляет нижний уровень иерархии телекоммуникационной сети. Основное назначение сети доступа — *концентрация информационных потоков*, поступающих по многочисленным каналам связи от оборудования клиентов, в сравнительно небольшом количестве узлов магистральной сети.

В случае компьютерной сети **термиナルным оборудованием** являются компьютеры (часто объединенные в локальную сеть), в случае телефонной — телефонные аппараты, подключенные к офисному телефонному коммутатору PBX (Private Branch Exchange).

Магистральная сеть объединяет отдельные сети доступа, обеспечивая транзит трафика между ними по высокоскоростным каналам.

ПРИМЕЧАНИЕ

Легко заметить, что любая национальная сеть автомобильных дорог имеет ту же иерархическую структуру, что и крупная телекоммуникационная сеть. Обычно существует разветвленная инфраструктура небольших дорог (аналог сети доступа), связывающих деревни и поселки. Эти дороги довольно узкие, так как интенсивность трафика между этими населенными пунктами невысокая и нет смысла делать подобные дороги многорядными. Такие дороги вливаются в более скоростные и соответственно более широкие дороги, которые, в свою очередь, имеют соединения с национальными супермагистралями (аналог магистральной сети).

Информационные центры, или **центры управления сервисами**, предоставляют информационные услуги сети. В таких центрах может храниться информация двух типов: пользовательская и вспомогательная служебная.

- ❑ Пользовательская информация — это информация, которая непосредственно интересует конечных пользователей сети. В компьютерных сетях примером информационных ресурсов такого типа могут служить веб-сайты, на которых расположена разнообразная справочная и новостная информация, информация электронных магазинов и т. п. В телефонных сетях подобные центры оказывают услуги экстренного вызова (например, милиции, скорой помощи) и справочные услуги различных организаций и предприятий — вокзалов, аэропортов, магазинов и т. п.
- ❑ Вспомогательная служебная информация помогает поставщику услуг предоставлять услуги пользователям, например, с помощью различных систем аутентификации и авторизации пользователей; организация, владеющая сетью, проверяет права пользователей на получение тех или иных услуг.

Большинство телекоммуникационных сетей включают в себя **системы управления сетью** (Network Management Systems, NMS), которые автоматизируют действия администраторов по мониторингу состояния сети и конфигурированию сетевых устройств и сервисов. Такие системы обычно представляют собой сложные программные комплексы, работающие на выделенных компьютерах.

Сети операторов связи

Специализированное предприятие, которое создает телекоммуникационную сеть для оказания общедоступных услуг (сервисов), владеет этой сетью и поддерживает ее работу, называется **оператором связи**.

Операторы связи осуществляют свою деятельность на коммерческой основе, заключая договоры с потребителями услуг. Особенностью современных операторов связи является то, что они, как правило, оказывают услуги нескольких типов, например услуги телефонии и доступа в Интернет.

Телефонные услуги включают в себя соединение двух абонентов, переадресацию вызовов, голосовую почту, доступ к справочным службам.

Услуги компьютерных сетей — это предоставление доступа в Интернет, электронная почта, информационные порталы, связывание локальных сетей и многое другое.

Некоторые услуги требуют совместного оперативного взаимодействия компьютерных и телефонных сетей. Ярким примером таких услуг является международная IP-телефония, которая отобрала у традиционной международной телефонии значительную часть клиентов.

Услуги можно разделить и по другому принципу — на **транспортные и информационные**. Телефонный разговор — это пример услуги первого типа, так как оператор доставляет голосовой трафик от одного абонента к другому. Примерами информационных услуг являются справочные услуги телефонной сети или веб-сайтов.

Именно этот тип принципа различия услуг отражается в названиях телекоммуникационных компаний. Мы говорим «оператор» применительно к традиционным компаниям, основным бизнесом которых всегда были телефонные услуги и услуги предоставления каналов связи в аренду, то есть транспортные услуги. Название **поставщик услуг, или провайдер**, стало популярно с массовым распространением Интернета и его информационной услуги WWW. Оборудование оператора связи — телефонные коммутаторы, маршрутизаторы и коммутаторы компьютерных сетей — размещается в так называемых **точках присутствия оператора** (Point of Presence, POP), которые представляют собой специально оборудованные технические центры.

Все множество клиентов — потребителей услуг телекоммуникационных сетей — можно разделить на два больших лагеря.

1. **Массовые (индивидуальные) клиенты**, для которых местом потребления услуг выступает квартира или частный дом. Таким клиентам нужны, прежде всего, базовые услуги — телефонная связь, телевидение, радио, доступ в Интернет.
2. **Корпоративные клиенты** — это предприятия и организации различного профиля, состоящие из нескольких территориально рассредоточенных отделений и филиалов, а также имеющие сотрудников, часто работающих дома, нуждаются в расширенном наборе услуг. Прежде всего, такой услугой является **виртуальная частная сеть** (Virtual Private Network, 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.

Пример сети оператора связи, обслуживающего клиентов разных типов, приведен на рис. 1.40.

В последнее время корпоративные пользователи все чаще получают не только транспортные, но и информационные услуги операторов, например, переносят собственные веб-сайты и базы данных на территорию оператора, поручая последнему поддерживать их работу и обеспечивать быстрый доступ к ним сотрудникам предприятия и, возможно, других пользователей сети оператора.



Рис. 1.40. Сеть оператора связи

Корпоративные сети

Корпоративная сеть – это сеть, главным назначением которой является поддержание работы конкретного предприятия, владеющего данной сетью. Пользователями корпоративной сети являются только сотрудники данного предприятия. В отличие от сетей операторов связи, корпоративные сети, в общем случае, не оказывают услуг сторонним организациям или пользователям.

Корпоративные сети, так же как и сети операторов связи, относятся к телекоммуникационным сетям, и, следовательно, структура корпоративной сети в целом соответствует рассмотренной ранее обобщенной структуре сети оператора связи (см. рис. 1.40). Однако имеются и различия. Например, названия структурных единиц корпоративной сети отражают не только территорию покрытия, но и организационную структуру предприятия. Так, принято делить корпоративную сеть на сети отделов и рабочих групп, сети зданий и кампусов, магистраль.

Хотя формально корпоративной сетью является сеть предприятия любого масштаба, сеть уровня отдела или здания редко называют корпоративной. Обычно это название используют для сети крупного предприятия, включающей большое количество сетей масштаба отдела и здания, расположенных в разных городах и объединенных глобальными связями.

Сети отделов – это сети, которые используются сравнительно небольшой группой сотрудников, работающих в одном отделе предприятия. Эти сотрудники решают некоторые общие задачи, например, ведут бухгалтерский

учет или занимаются маркетингом. Считается, что отдел может насчитывать до 100–150 сотрудников. Сеть отдела — это локальная сеть, которая охватывает все помещения, принадлежащие отделу. Это могут быть несколько комнат или этаж здания.

Главной целью сети отдела является разделение локальных ресурсов, таких как приложения, данные, лазерные принтеры и модемы. Обычно сети отделов не подразделяются на подсети, имея в своем составе один или два файловых сервера. В этих сетях локализуется большая часть трафика предприятия. Локальная сеть, показанная на рис. 1.36, является примером сети масштаба отдела.

Задачи сетевого администрирования на уровне отдела относительно просты: добавление новых пользователей, устранение простых отказов, установка новых узлов и новых версий программного обеспечения. Такой сетью может управлять сотрудник, посвящающий обязанностям администратора только часть своего времени.

Сеть отдела может входить в состав сети здания (кампуса) или же представлять собой сеть удаленного офиса предприятия. В первом случае сеть отдела подключается к сети здания или кампуса с помощью технологии локальной сети, которой сегодня, скорее всего, будет одна из представительниц семейства Ethernet. Во втором случае сеть удаленного офиса подключается непосредственно к магистрали сети с помощью одной из технологий глобальных сетей, например Frame Relay.

Сеть здания (кампуса) объединяет сети различных отделов одного предприятия в пределах отдельного здания или в пределах одной территории (кампуса), покрывающей площадь в несколько квадратных километров. Для построения сетей зданий (кампусов) используются технологии локальных сетей, возможностей которых достаточно, чтобы обеспечить нужную зону покрытия.

Услуги такой сети включают взаимодействие между сетями отделов, доступ к общим базам данных предприятия, доступ к общим факс-серверам, высокоскоростным модемам и высокоскоростным принтерам. В результате сотрудники каждого отдела предприятия получают доступ к некоторым файлам и ресурсам сетей других отделов. Важной услугой, предоставляемой сетями кампусов, является доступ к корпоративным базам данных независимо от того, на каких типах компьютеров эти базы располагаются.

Именно на уровне сети кампуса возникают проблемы интеграции неоднородного аппаратного и программного обеспечения. Типы компьютеров, сетевых операционных систем и сетевого аппаратного обеспечения могут различаться в каждом отделе. Отсюда вытекают сложности, связанные с управлением сетями кампусов.

Сети масштаба предприятия, или корпоративные сети, прежде всего, характеризует *масштабность*.

Число пользователей и компьютеров в корпоративной сети может измечаться тысячами, а число серверов — сотнями; расстояния между сетями отдельных территорий могут оказаться такими, что использование глобальных связей становится необходимым. Поэтому корпоративная сеть является составной сетью, которую можно представить в виде «островков» локальных сетей, «плавающих» в телекоммуникационной среде. Для соединения удаленных локальных сетей и отдельных компьютеров в корпоративной сети применяются разнообразные телекоммуникационные средства, в том числе каналы первичных сетей, радиоканалы, спутниковая связь.

Корпоративные сети отличаются также тем, что в них на первый план выходят *информационные услуги*, они не могут ограничиться только транспортными услугами. Если сети операторов связи могут и не предоставлять информационных услуг, так как компьютеры пользователей находятся за пределами зоны их ответственности, то корпоративные сети не могут себе этого позволить. Настольные компьютеры пользователей и серверы являются неотъемлемой частью любой корпоративной сети, поэтому разработчики и специалисты по обслуживанию корпоративных сетей должны это учитывать.

Непременным атрибутом такой сложной и крупномасштабной сети является *высокая степень неоднородности* (гетерогенности) — нельзя удовлетворить потребности тысяч пользователей с помощью однотипных программных и аппаратных средств. В корпоративной сети обязательно используются различные типы компьютеров — от мейнфреймов до персональных компьютеров, несколько типов операционных систем и множество различных приложений. Неоднородные части корпоративной сети должны работать как единое целое, предоставляя пользователям по возможности удобный и простой доступ ко всем необходимым ресурсам.

Появление корпоративных сетей — это хорошая иллюстрация известного философского постулата о *переходе количества в качество*. При объединении в единую сеть отдельных сетей крупного предприятия, имеющего филиалы в разных городах и даже странах, многие количественные характеристики объединенной сети превосходят некоторый критический порог, за которым начинается новое качество. В этих условиях существующие методы и подходы к решению традиционных задач в сетях меньших масштабов для корпоративных сетей оказываются непригодными. На первый план выходят такие задачи и проблемы, которые в сетях рабочих групп, отделов и даже кампусов либо имели второстепенное значение, либо вообще не проявлялись.

Примером может служить простейшая (для небольших сетей) задача — ведение учетных данных о пользователях сети. Наиболее простой способ ее решения — помещение учетных данных всех пользователей в локальную базу учетных данных каждого компьютера, к ресурсам которого эти пользователи должны иметь доступ. При попытке доступа данные извлекаются из локальной учетной базы, и на их основе доступ предоставляетя или не предоставляется. Для небольшой сети, состоящей из 5–10 компьютеров, этот подход

работает очень хорошо. Но если в сети насчитывается несколько тысяч пользователей, каждому из которых нужен доступ к нескольким десяткам серверов, очевидно, что это решение становится крайне неэффективным. Администратор должен повторить несколько десятков раз (по числу серверов) операцию занесения учетных данных каждого пользователя. Сам пользователь также вынужден повторять процедуру логического входа каждый раз, когда ему нужен доступ к ресурсам нового сервера. Хорошее решение этой проблемы для крупной сети — применение централизованной справочной системы, в базе данных которой хранятся учетные записи всех пользователей сети. Администратор один раз выполняет операцию занесения данных пользователя в эту базу, а пользователь один раз выполняет процедуру логического входа, причем не на отдельный сервер, а в сеть целиком.

При переходе от более простого типа сетей к более сложному — от сетей отдела к корпоративной сети — географические расстояния увеличиваются, поддержание связи компьютеров становится все более сложным и дорогостоящим. По мере увеличения масштабов сети повышаются требования к ее надежности, производительности и функциональным возможностям. По сети циркулируют все возрастающие объемы данных, и сеть должна обеспечивать их безопасность и защищенность наряду с доступностью. Все это приводит к тому, что корпоративные сети строятся на основе наиболее мощного и разнообразного оборудования и программного обеспечения.

Стандартизация сетей

Многоуровневый подход

Организация взаимодействия между устройствами сети является сложной задачей. Для решения сложных задач 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 внутреннего устройства и концентрируя внимание на его функциях и способе взаимодействия с другими модулями. В результате такого логического упрощения задачи появляется возможность независимого тестирования, разработки и модификации модулей. Так, любой из показанных на рис. 1.41 модулей может быть переписан заново. Пусть, например, это будет модуль A, и если при этом разработчики усовершенствуют его внутреннюю организацию, но сохранят без изменения межмодульные интерфейсы A—B, A—F 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. 1.42).

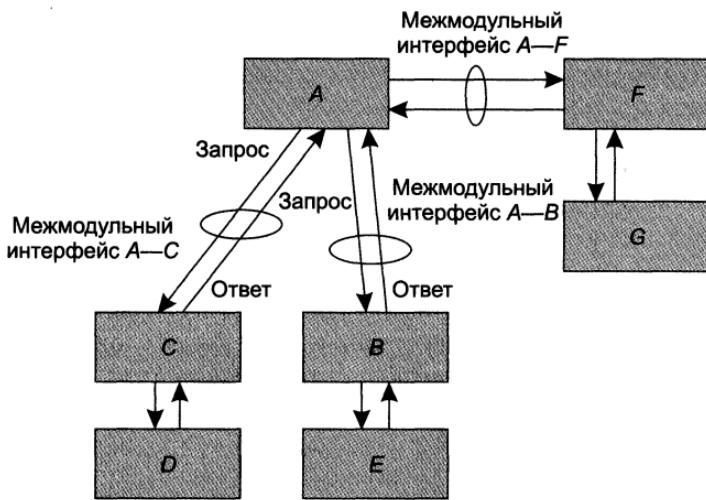


Рис. 1.41. Пример декомпозиции задачи на функциональные модули

Каждый вышестоящий уровень использует нижестоящий в качестве удобного инструмента для решения своих задач. Так, группа модулей, находящихся на верхнем уровне иерархии, может обращаться с запросами на выполнение тех или иных функций только к модулям непосредственно прилегающего нижнего уровня 2, а модули уровня 2, в свою очередь, могут обращаться за услугами к модулям уровня 1. В то же время 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 каждого уровня.

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

Задача организации взаимодействия компьютеров в сети тоже может быть представлена в виде иерархически организованного множества модулей. 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, работающих на разных компьютерах.

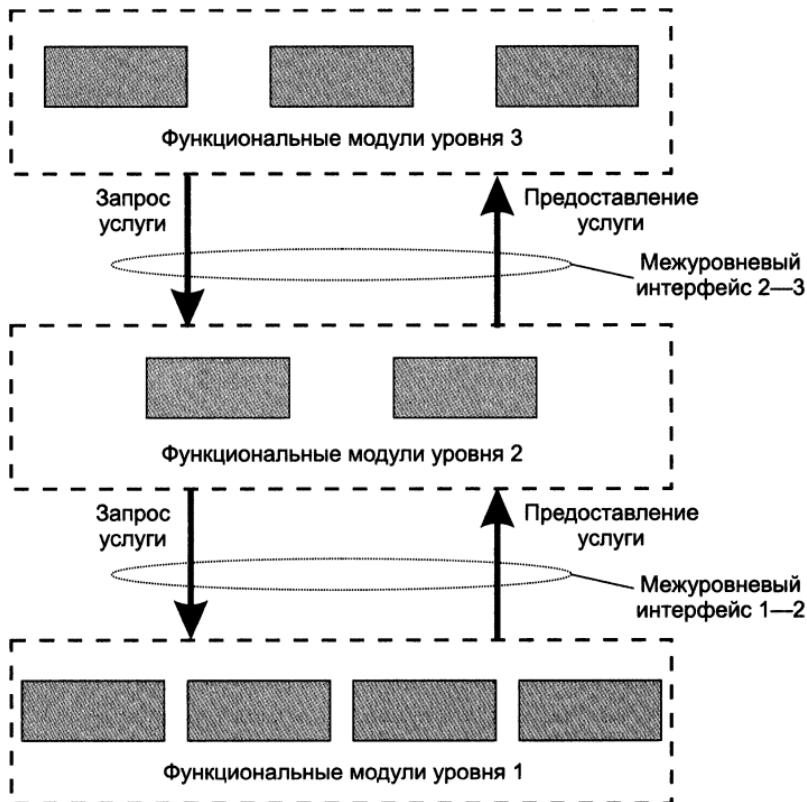


Рис. 1.42. Многоуровневый подход — создание иерархии задач

Оба участника сетевого обмена должны принять множество соглашений на каждом уровне взаимодействия. Например, они должны согласовать уровни и форму электрических сигналов, перечень сообщений и их формат, 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.

Собственно, мы уже прибегали к этому приему, когда изучали простейшую сеть, состоящую из двух компьютеров и позволяющую пользователям совместно применять принтер (см. рис. 1.9). Давайте вспомним, как при решении этой задачи в нашем примере функции разделялись между тремя уровнями сетевых средств. Верхний уровень этой иерархии, организующий диалог клиента и сервера печати, в качестве «инструмента» для передачи своих сообщений через сеть использует средства нижележащего уровня драйверов межмашинной связи (MMC). В свою очередь, уровень MMC, решая порученную ему задачу, время от времени обращается к услугам более низкого уровня физической передачи, который выполняет для него передачу битов по кабелю.

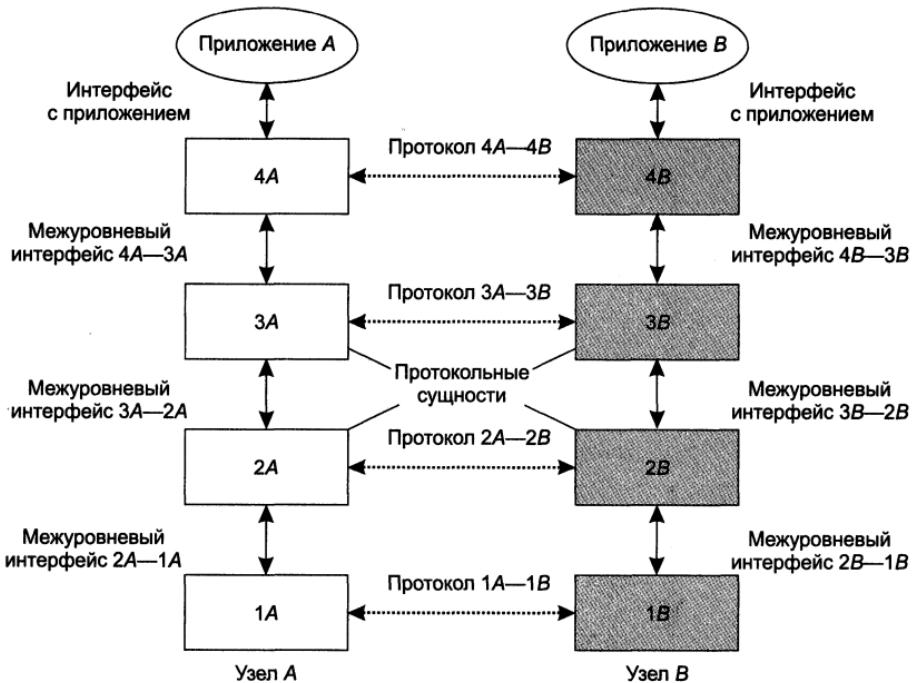


Рис. 1.43. Протоколы и интерфейсы многоуровневых средств взаимодействия двух узлов

Рисунок 1.43 поясняет основные идеи и термины, касающиеся многоуровневой организации сетевых средств на примере взаимодействия двух сетевых узлов, *A* и *B*. С каждой стороны средства сетевого взаимодействия представлены четырьмя уровнями (количество уровней в этом примере условно). Каждый уровень (кроме самого нижнего) поддерживает *три* интерфейса. Во-первых, это два интерфейса услуг с выше- и нижележащим уровнями «своей» иерархии средств. Во-вторых, это интерфейс со средствами взаимодействия другой стороны, расположенным на том же уровне иерархии. С последним типом интерфейса мы уже неоднократно встречались, используя для его обозначения термин «протокол».

Протоколы нижних уровней часто реализуются комбинацией программных и аппаратных средств, а протоколы верхних уровней, как правило, программными средствами.

ПРИМЕЧАНИЕ

В сущности, термины «протокол» и «интерфейс» выражают одно и то же понятие — формализованное описание процедуры взаимодействия двух объектов, но традиционно в сетях за ними закрепили разные области действия: протокол обычно определяет правила взаимодействия модулей одного уровня в разных узлах, а интерфейс — правила взаимодействия модулей соседних уровней в одном узле.

Модули, поддерживающие протокол, а также межуровневые интерфейсы, называют **протокольными сущностями** или, для краткости, тоже **протоколами**.

Протокольные сущности одного уровня двух взаимодействующих сторон обмениваются сообщениями в соответствии с определенным для них протоколом. Сообщения состоят из заголовка и поля данных (иногда оно может отсутствовать). Обмен сообщениями является своеобразным языком общения, с помощью которого каждая из сторон «объясняет» другой стороне, что необходимо сделать на каждом этапе взаимодействия. Работа каждого протокольного модуля заключается 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. Протокольные сущности одного уровня не общаются между собой непосредственно, в этом общении всегда участвуют посредники — средства протоколов нижележащих уровней. И только самые нижние уровни двух иерархий взаимодействуют напрямую.

Многоуровневый набор протоколов, достаточный для организации взаимодействия узлов в сети, называется **стеком коммуникационных протоколов**.

Понятно, что для одного и того же протокола (протокольной сущности) может существовать сколь угодно много различных его реализаций. Эти реализации могут отражать строгость следования разработчика описанию протокола или же включать некоторые дополнительные функции, которые не были описаны в спецификации. И даже в том случае, когда с функциональной точки зрения реализации протокола идентичны, они могут различаться эффективностью программной реализации, например, одна реализация может требовать меньше памяти или процессорного времени, чем другая. 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.

Модель OSI

К концу 70-х годов в мире уже существовало большое разнообразие в стеках коммуникационных протоколов, среди которых можно было назвать, например, такие популярные стеки, как DECnet (фирменный стек компании Digital), TCP/IP (стек Интернета) и SNA (фирменный стек компании IBM). Однако компьютеры и другие сетевые устройства, поддерживающие разные стеки протоколов, будучи помещенными в одну сеть, «отказывались» работать друг с другом. Одним из путей преодоления такой несовместимости

в то время виделся всеобщий переход на единый стек протоколов, который аккумулировал бы в себе все лучшее, что было в других, уже существующих стеках. Именно с таких академических позиций несколько международных организаций по стандартизации¹ подошли к разработке нового стека коммуникационных протоколов. Важнейшим результатом их работы стало создание **стандартной модели взаимодействия открытых систем** (Open System Interconnection, OSI).

ПРИМЕЧАНИЕ

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.

Назначение модели OSI состоит в обобщенном стандартном представлении средств сетевого взаимодействия для сетей с коммутацией пакетов. Она разрабатывалась в качестве своего рода универсального языка сетевых специалистов, именно поэтому ее называют также **справочной моделью**.

Модель OSI определяет:

- уровни взаимодействия систем в сетях с коммутацией пакетов;
- стандартные названия уровней;
- функции, которые должен выполнять каждый уровень.

Модель OSI не содержит описаний конкретных протоколов и их реализаций.

Модель OSI делит средства взаимодействия на *семь уровней*, за которыми закреплены названия (рис. 1.44):

- прикладной;
- представления;
- сеансовый;
- транспортный;
- сетевой;
- канальный;
- физический.

Модель OSI описывает только *системные* средства взаимодействия, реализуемые операционной системой, системными утилитами, системными аппаратными средствами. Модель не включает средства взаимодействия приложений конечных пользователей. Приложения могут обращаться к системным средствам сетевого взаимодействия, используя специально разработанный для этих целей набор стандартных процедур операционной системы — **прикладной программный интерфейс** (Application Program Interface, API).

¹ В частности, в число этих организаций входили International Organization for Standardization (ISO), часто называемая еще International Standards Organization, а также International Telecommunications Union (ITU) и некоторые другие.

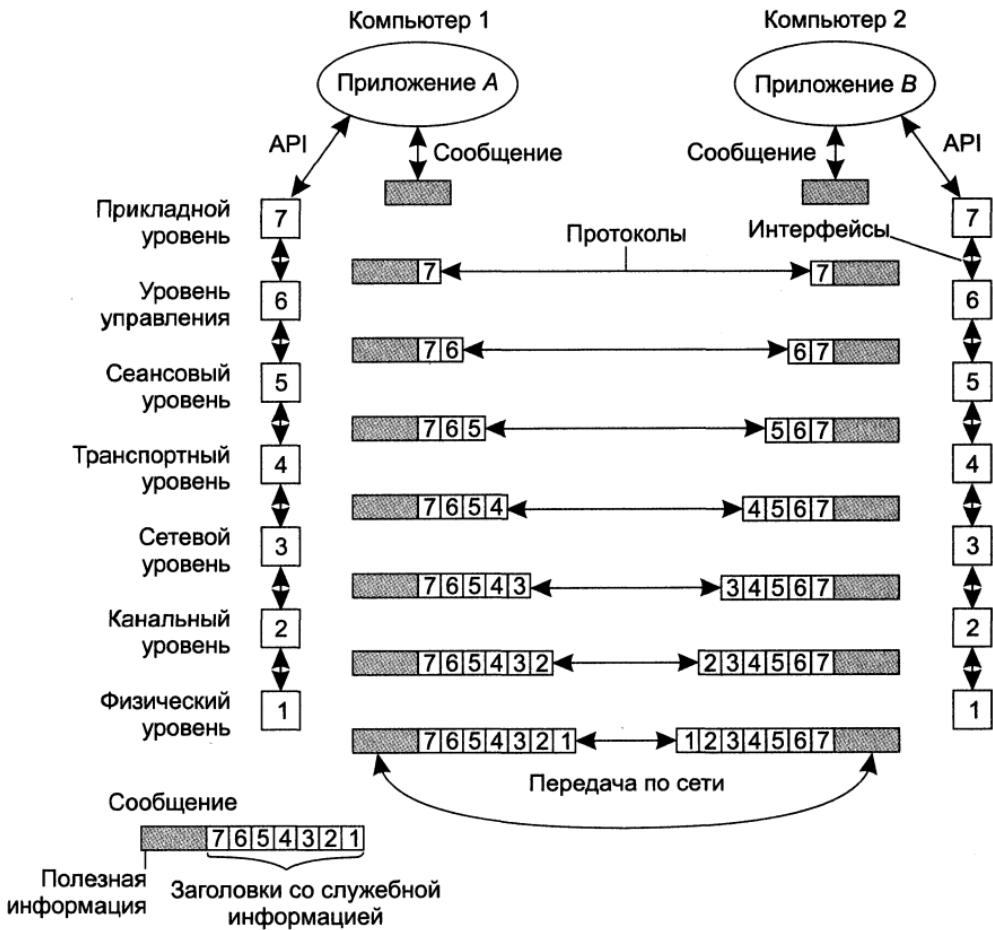


Рис. 1.44. Модель взаимодействия открытых систем OSI

Итак, пусть, например, приложению A требуется передать файл приложению B. Для этого приложение A посылает сообщение-запрос клиенту файловой службы, которая, в соответствии с разделением функций в модели OSI, относится к прикладному уровню¹. На основании этого запроса клиент в соответствии с принятым протоколом формирует сообщение для передачи серверу файловой службы, также работающему на прикладном уровне другого компьютера. Формат сообщения предусматривает наличие поля заголовка прикладного уровня (на рисунке он обозначен цифрой 7), в котором клиент размещает управляющую информацию, адресованную файловому серверу, например, полное имя файла в файловой системе сервера и тип операции «открыть файл». Но для того, чтобы доставить это сообщение по назначе-

¹ Важно различать уровень взаимодействия приложений и прикладной уровень семиуровневой модели.

нию, предстоит решить еще много задач, ответственность за которые несут нижележащие уровни.

Поэтому прикладной уровень, используя *межуровневый интерфейс*, передает запрос на выполнение необходимых для него действий расположенному ниже уровню представления. Запрос включает сформированное на прикладном уровне сообщение, а также некоторую дополнительную служебную информацию, без которой нижележащий уровень не может выполнить указанные действия, например, расшифровать текст, который хранится на сервере в зашифрованном виде. Протокол уровня представления выполняет требуемые операции и добавляет к сообщению собственную служебную информацию — заголовок уровня представления, в котором содержатся указания для протокола уровня представления компьютера-адресата, например тип шифра. Полученное в результате сообщение передается вниз сеансовому уровню, который, в свою очередь, добавляет свой заголовок и т. д. Наконец, сообщение достигает нижнего, физического уровня, который собственно и передает его по линиям связи компьютеру-адресату¹. К этому моменту сообщение «обрастает» заголовками всех уровней.

Физический уровень помещает сообщение на физический выходной интерфейс компьютера 1, и оно начинает свое «путешествие» по сети.

Когда сообщение по сети поступает на входной интерфейс компьютера 2, оно принимается его физическим уровнем и последовательно перемещается вверх с уровня на уровень. Каждый уровень анализирует и обрабатывает заголовок своего уровня, выполняя соответствующие функции, а затем удаляет этот заголовок и передает сообщение вышележащему уровню.

ПРИМЕЧАНИЕ

Мы описали упрощенную схему взаимодействия протокольных сущностей разных уровней, в реальности взаимодействие может выглядеть сложнее. Типичной, например, является ситуация, когда для отработки запроса «сверху» некоторому протокольному модулю приходится посыпать несколько собственных сообщений модулю того же уровня узла-напарника через сеть, прежде чем он сможет передать оригинальное сообщение вышестоящего уровня нижестоящему.

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**.

¹ На физическом уровне заголовок отсутствует, если не считать в качестве такового стартовый сигнал при передаче очередного байта.

Функции уровней модели OSI

А сейчас давайте остановимся на том, какие функции выполняет каждый уровень модели OSI.

Прикладной уровень (application layer) — это в действительности просто набор разнообразных протоколов, с помощью которых пользователи сети получают доступ к разделяемым ресурсам, таким как файлы, принтеры или гипертекстовые веб-страницы, а также организуют свою совместную работу, например, по протоколу электронной почты. Единица данных, которой оперирует прикладной уровень, обычно называется **сообщением**.

Уровень представления (presentation layer), как явствует 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.

Сеансовый уровень (session layer) обеспечивает управление 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, и он редко реализуется в виде отдельных протоколов. Функции этого уровня часто объединяют с функциями прикладного уровня и реализуют в одном протоколе.

Транспортный уровень (transport layer) обеспечивает приложениям или 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.

Выбор класса сервиса транспортного уровня определяется, с одной стороны, тем, в какой степени задача обеспечения надежности решается самими приложениями и протоколами более высоких, чем транспортный, уровней. С другой стороны, этот выбор зависит от того, насколько надежной является

система транспортировки данных в сети, обеспечивающая уровнями, расположеннымными ниже транспортного — сетевым, канальным и физическим. Так, если качество каналов передачи связи очень высокое и вероятность возникновения ошибок, не обнаруженных протоколами более низких уровней, невелика, то разумно воспользоваться одним из облегченных сервисов транспортного уровня, не обремененных многочисленными проверками, квитированием и другими приемами повышения надежности. Если же транспортные средства нижних уровней очень ненадежны, то целесообразно обратиться к наиболее развитому сервису транспортного уровня, который использует максимум средств обнаружения и устранения ошибок, включая предварительное установление логического соединения, контроль доставки сообщений по контрольным суммам и циклической нумерации пакетов, установление тайм-аутов доставки и т. п.

Сетевой уровень (*network layer*) служит для образования единой транспортной системы, объединяющей несколько сетей, называемой составной сетью. Решение этой задачи возлагается на конечные узлы и маршрутизаторы и включает решение следующих частных задач:

- определение маршрута через составную сеть от узла-отправителя до узла-получателя;
- организацию продвижения данных по этому маршруту;
- согласование технологий при передаче данных из сети, построенной на одной технологии, в сеть, построенную на другой технологии;
- управление параметрами процесса передачи данных, такими как временные задержки, уровни загрузки линий связи и др.;
- создание надежных и гибких барьеров на пути нежелательного трафика между сетями.

Данные, которые необходимо передать через составную сеть, поступают на сетевой уровень от вышестоящего транспортного уровня. Эти данные снабжаются заголовком сетевого уровня, образуя **пакет** — так называется PDU сетевого уровня.

ПРИМЕЧАНИЕ

В заключение отметим, что на сетевом уровне определяются два вида протоколов. Первый вид — маршрутизуемые протоколы — реализует продвижение пакетов через сеть. Именно эти протоколы обычно имеют в виду, когда говорят о протоколах сетевого уровня. Однако часто к сетевому уровню относят и другой вид протоколов, называемых маршрутизирующими протоколами, или протоколами маршрутизации. С помощью этих протоколов маршрутизаторы собирают информацию о топологии межсетевых соединений, на основании которой осуществляется выбор маршрута продвижения пакетов.

Канальный уровень (*data link layer*) обеспечивает прозрачность соединения для сетевого уровня. Для этого он предлагает ему следующие услуги:

- установление логического соединения между взаимодействующими узлами;
- согласование в рамках соединения скоростей передатчика и приемника информации;
- обеспечение надежной передачи, обнаружение и коррекция ошибок.

Для решения этих задач канальный уровень формирует из пакетов собственные протокольные единицы данных — **кадры**, состоящие из поля данных и заголовка. Канальный уровень помещает пакет в поле данных одного или нескольких кадров и заполняет собственной служебной информацией заголовок кадра.

Рассмотрим, например, как канальный уровень решает задачу обеспечения надежности передачи. Для этого канальный уровень фиксирует границы кадра, помещая специальную последовательность битов в его начало и конец, а затем добавляет к кадру контрольную сумму. Контрольная сумма вычисляется по некоторому алгоритму как функция от всех байтов кадра. На стороне получателя канальный уровень группирует биты, поступающие с физического уровня, в кадры, снова вычисляет контрольную сумму полученных данных и сравнивает результат с контрольной суммой, переданной в кадре. Если они совпадают, кадр считается правильным. Если же контрольные суммы не совпадают, фиксируется ошибка. В функции канального уровня входят не только обнаружение ошибок, но и исправление их за счет повторной передачи поврежденных кадров. Однако эта функция не является обязательной и в некоторых реализациях канального уровня она отсутствует, например в Ethernet.

В сетях, построенных на основе разделяемой среды, физический уровень выполняет еще одну функцию — проверяет доступность разделяемой среды. Эту функцию иногда выделяют в отдельный подуровень **управления доступом к среде** (Medium Access Control, MAC).

Протокол канального уровня обычно работает в пределах сети, являющейся одной из частей более крупной составной сети, объединенной протоколами сетевого уровня. Адреса, с которыми работает протокол канального уровня, используются для доставки кадров только в пределах этой сети, а для перемещения пакетов между сетями применяются адреса уже следующего, сетевого, уровня.

Протоколы канального уровня реализуются как на конечных узлах (средствами сетевых адаптеров и их драйверов), так и на всех промежуточных сетевых устройствах.

Физический уровень (physical layer) поддерживает интерфейс с канальным уровнем. На стороне отправителя физический уровень получает с канального уровня кадры, рассматривая их как неструктурированный поток битов, которые он должен передать по физическим каналам связи, таким как коаксиальный кабель, витая пара, оптоволоконный кабель или цифровой

территориальный канал. Протокол физического уровня представляет биты данных в виде электрических импульсов и пытается передать их, по возможности без искажений, в соответствии с принятыми в данном протоколе параметрами электрических сигналов (уровнями напряжения, соответствующими 1 и 0, тактовой частотой и т. п.). Любое устройство для подключения к сети должно иметь средства реализации функций физического уровня. Со стороны компьютера функции физического уровня выполняются сетевым адаптером, в коммутаторах и маршрутизаторах — это функции физических интерфейсов.

Распределение функций между различными элементами сети

Функциональность стека протоколов в целом может быть востребована только конечными узлами, а коммуникационные устройства — маршрутизаторы и коммутаторы, — решающие задачу транспортировки сообщений между конечными узлами, ограничиваются поддержкой функциональности трех нижних уровней.

На рис. 1.45 показаны основные элементы компьютерной сети: конечные узлы — компьютеры и промежуточные узлы — коммутаторы и маршрутизаторы. Маршрутизаторы служат пограничными устройствами, разделяя составную сеть на подсети.

Коммутаторы обычно поддерживают функции двух нижних уровней, физического и канального, что ограничивает их возможности передачей данных в пределах только одной подсети. Однако некоторые коммутаторы, работающие на основе технологии виртуальных каналов, могут поддерживать как два уровня протоколов, так и три.

Маршрутизаторы поддерживают функции всех трех нижних уровней, так как сетевой уровень нужен им для объединения подсетей различных технологий в составную сеть и нахождения маршрута между конечными узлами через составную сеть, а функции нижних уровней — для передачи данных в пределах отдельных подсетей.

Компьютеры, на которых работают сетевые приложения, в общем случае поддерживают функции всех уровней. Протоколы прикладного уровня, пользуясь сервисами протоколов уровня представления и сеансового уровня, предоставляют приложениям набор сетевых услуг в виде сетевого 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.

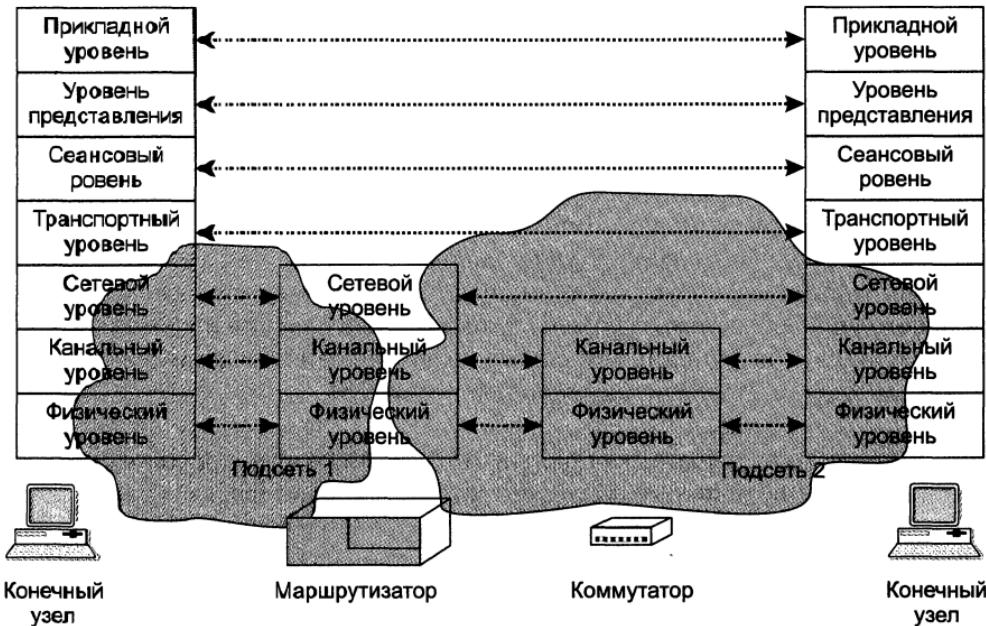


Рис. 1.45. Распределение функций между различными элементами сети

ПРИМЕЧАНИЕ

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 коммутаторы часто поддерживают протоколы SNMP и telnet, которые не нужны для выполнения основных функций этих устройств, но позволяют конфигурировать их и управлять ими удаленно. Все эти протоколы являются протоколами прикладного уровня и выполняют некоторые вспомогательные (служебные) функции транспортной системы.

В компьютерах коммуникационные протоколы всех уровней (кроме физического и части функций канального уровня) реализуются программно операционной системой или системными приложениями.

Стандартные стеки протоколов

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.

Важнейшим направлением стандартизации в области вычислительных сетей является стандартизация коммуникационных протоколов. Наиболее известными стандартными стеками протоколов являются: OSI, TCP/IP, IPX/SPX, NetBIOS/SMB, DECnet, SNA (не все из них применяются сегодня на практике).

В большинстве этих популярных стеков протоколов нет полного соответствия рекомендациям модели OSI, в частности, ни один из них не разбит на семь уровней. Чаще всего в стеке явно выделяются 3–4 уровня: уровень сетевых адаптеров, в котором реализуются протоколы физического и канального уровней, сетевой уровень, транспортный уровень и уровень служб, включающий в себя функции сеансового уровня, уровня представления и прикладного уровня. Исключение составляет **стек OSI**.

Следует четко различать два понятия **модель OSI** и **стек протоколов OSI**. Модель OSI представляет собой абстрактную концепцию, а стек протоколов OSI, который построен на основе модели OSI, содержит набор описаний спецификаций конкретных протоколов, а также многочисленные программные реализации этих протоколов.

В отличие от других стеков протоколов, стек OSI полностью соответствует модели OSI, включая спецификации протоколов для всех семи уровней взаимодействия, определенных в этой модели. Это и понятно, разработчики стека OSI использовали модель OSI как прямое руководство к действию.

Особое место в ряду стандартных стеков протоколов занимает **стек TCP/IP**. Он был разработан по инициативе министерства обороны США (Department of Defense, DoD) более 30 лет назад для связи экспериментальной сети ARPAnet с другими сетями как набор общих протоколов для разнородной вычислительной среды. Сегодня этот стек используется для связи компьютеров в Интернете, а также в огромном числе корпоративных сетей. Так как стек TCP/IP был разработан до появления модели OSI, то, хотя он также имеет многоуровневую структуру, соответствие уровней стека TCP/IP уровням модели OSI достаточно условно. Мы будем подробно изучать протоколы этого стека в главе 3.

Стандарты, определяющие работу Интернета, носят особое название — **темы для обсуждения** (Request For Comments, RFC), что показывает гласный и открытый характер принимаемых стандартов. Ввиду постоянно растущей популярности Интернета RFC-документы становятся международными стандартами де-факто, многие из которых затем приобретают статус официальных международных стандартов. В соответствии с принципом открытости Интернета все RFC-документы находятся в свободном доступе. Список RFC-документов можно найти, в частности, на сайте www.rfc-editor.org.

ГЛАВА 2. Технологии локальных сетей

Вы спрашиваете, придет ли что-нибудь когда-нибудь на смену Ethernet? Конечно! Но вся прелест состоит в том, что до сих пор, каждый раз, когда появлялось что-то на замену Ethernet, люди, ответственные за продвижение новой технологии, снова выбирали для нее название Ethernet.

*Роберт Мэткалф
(изобретатель Ethernet)*

Сегодня технологии локальных сетей описывать проще, чем, скажем, 10–15 лет назад, так как в них безраздельно господствует одна технология — Ethernet. Остальные технологии, такие как Arcnet, Token Ring и FDDI, остались в прошлом несмотря на то, что они обладали хорошими техническими характеристиками и имели многочисленных пользователей. Неизвестно, что больше повлияло на такую ситуацию, то ли предельная простота технологии, а значит, и низкая стоимость оборудования Ethernet и его эксплуатации, то ли удачное имя, то ли просто необыкновенное везение, как считает изобретатель этой технологии, но факт остается фактом — локальные сети стали однородными.

Чтобы соответствовать разнообразию потребностей локальных сетей, Ethernet как единственная технология этого класса также должна быть разнообразной. И современная технология Ethernet удовлетворяет этому требованию — в отношении как скорости, так и поддержки всех популярных сред передачи данных. В этой главе даются краткие справочные сведения для каждой из иерархии скоростей Ethernet, покрывающей сегодня диапазон от 10 Мбит/с до 10 Гбит/с и со скоростью 100 Гбит/с «на горизонте», а также для каждого из вариантов физической среды, включающих коаксиал, витую пару, оптическое волокно и радиоволны.

В этой главе мы рассмотрим технологии как проводных, так и беспроводных локальных сетей. При изучении проводных сетей основной акцент будет сделан на коммутируемой версии Ethernet, но перед ее изучением мы кратко изучим основные принципы работы Ethernet на разделяемой среде, так как без этих знаний трудно понять общую структуру стандартов и применяемую

терминологию, которые в свое время были разработаны в расчете на разделяемые среды. Кроме того, разделяемая среда по-прежнему остается основным средством организации беспроводных локальных сетей.

Описание Ethernet не исчерпывается данной главой, в главе 4, посвященной глобальным сетям, рассматриваются основные принципы новой версии Стандарт Ethernet, которая была разработана для работы в сетях операторов связи.

Особенности локальных сетей

Локальные сети являются неотъемлемой частью любой современной компьютерной сети. Если мы рассмотрим структуру глобальной сети, например, Интернета или крупной корпоративной сети, то обнаружим, что практически все информационные ресурсы этой сети сосредоточены в локальных сетях, а глобальная сеть является транспортом, соединяющим многочисленные локальные сети.

Одним из основных назначений локальной сети является объединение компьютеров в пределах одного здания или нескольких близко стоящих зданий с целью предоставления пользователям сети доступа к информационным ресурсам и услугам локальных серверов.

Кроме того, локальные сети являются удобным средством группирования компьютеров для объединения их в глобальную сеть, так как в глобальной сети данные проще маршрутизировать между сетями, а не отдельными компьютерами. Примером могут служить беспроводные локальные сети, обслуживающие аэропорты или вокзалы. Они применяются, как правило, не для обмена информацией между временными пользователями такой сети, а для доступа этих пользователей в Интернет, причем доступ организуется не для каждого отдельного пользователя, а для локальной сети в целом.

Локальные сети применяются также в телекоммуникационных сетях другого типа, например в телефонных сетях. Так, системы управления телефонными коммутаторами или первичными сетями обычно строятся на основе локальной сети, которая объединяет компьютеры ее операторов и обеспечивает им доступ к устройствам управления, встроенным в оборудование телекоммуникационной сети.

Технологии локальных сетей прошли большой путь. Практически во всех технологиях 80-х годов использовалась *разделяемая среда* как удобное и экономичное средство объединения компьютеров на физическом уровне. С середины 90-х в локальных сетях стали также применяться *коммутируемые* версии технологий, в которых вместо разделяемой среды применяются коммутаторы, работающие в соответствии с общими принципами коммутации пакетов, рассмотренными в главе 1. Ведущей организацией по стандартизации технологий локальных сетей является IEEE (The Institute of Electrical and Electronics Engineers), точнее, ее комитет с кодовым названием 802.

Коммутируемые локальные сети обладают многими преимуществами по сравнению с сетями на разделяемой среде, наиболее важными из которых являются более высокие показатели производительности, надежности и масштабируемости. Основным препятствием на пути применения коммутаторов в локальных сетях была их относительно высокая стоимость, но уже к середине 90-х годов за счет прогресса в технологии цены коммутаторов существенно снизились, что привело к полному вытеснению разделяемой среды из проводных (кабельных) локальных сетей.

Достаточно долгое время в локальных сетях сосуществовало большое количество технологий: ArcNet, Ethernet, Token Ring, FDDI, 100VG-AnyLAN. Однако в середине 90-х годов начался процесс вытеснения технологией Ethernet всех остальных технологий локальных сетей. К сегодняшнему дню этот процесс практически завершился, производители сняли с производства оборудование Token Ring и других технологий, оставив администраторам и немногочисленным пользователям этих унаследованных сетей шанс поддерживать их работоспособность за счет рынка подержанных концентраторов и сетевых адаптеров.

Техника виртуальных локальных сетей (Virtual LAN, VLAN), которая появилась почти одновременно с первыми коммутаторами, позволила гибко и эффективно структурировать локальную сеть, разделяя ее на независимые сегменты. Такая виртуальная структуризация оказалась идеальным решением для создания крупных локальных сетей, в которых отдельные виртуальные сегменты объединяются в общую сеть маршрутизаторами.

Несмотря на успех коммутируемых локальных сетей, техника построения локальных сетей на разделяемой среде полностью не исчезла — она по-прежнему востребована в беспроводных локальных сетях, где радиоэфир естественным образом разделяется между всеми сетевыми адаптерами, находящимися в зоне приема.

Недавно технология Ethernet сделала еще один шаг в своем развитии, который может оказаться даже более революционным, чем отказ от разделяемой среды и появление виртуальных локальных сетей. Эта инициатива получила название Carrier Ethernet, это отражает тот факт, что технология Ethernet начала использоваться в глобальных сетях операторов связи. Правда, для этого технологию пришлось усовершенствовать, добавив такие важные функции, как мониторинг соединений, отделение адресов оператора от адресов пользователя, поддержка отказоустойчивости и ряд других. Если Carrier Ethernet закрепится в глобальных сетях, то это еще больше повысит однородность сетей и еще раз подтвердит мнение Роберта Меткалфа о том, что почти любая новая технология получает счастливое название Ethernet.

Локальные сети на разделяемой среде

Основная цель, которую ставили перед собой разработчики первых локальных сетей во второй половине 70-х годов, заключалась в нахождении простого и дешевого решения, позволяющего объединить в вычислительную

сеть несколько десятков компьютеров, находящихся в пределах одного здания. Решение должно было быть недорогим, поскольку в сеть объединялись недорогие компьютеры — появившиеся и быстро распространявшиеся тогда мини-компьютеры стоимостью 10 000–20 000 долларов. Количество их в одной организации было небольшим, поэтому предел в несколько десятков компьютеров представлялся вполне достаточным для практически любой локальной сети.

Для упрощения и, соответственно, удешевления аппаратных и программных решений разработчики первых локальных сетей остановились на совместном использовании, или **разделении**, среды передачи данных.

Этот метод связи компьютеров впервые был опробован при создании радиосети ALOHA Гавайского университета в начале 70-х под руководством Нормана Абрамсона (Norman Abramson). Сеть ALOHA работала по методу случайного доступа, когда каждый узел мог начать передачу пакета в любой момент времени. Если после этого он не дождался подтверждения приема в течение определенного тайм-аута, он посыпал 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 (Robert Metcalfe) повторил идею разделяемой среды сети ALOHA уже для проводного варианта технологии LAN. Непрерывный сегмент коаксиального кабеля стал аналогом общей радиосреды. Все компьютеры присоединялись к этому сегменту кабеля, поэтому при передаче сигналов одним из передатчиков все приемники получали один и тот же сигнал, как и при использовании радиоволн. Технология получила название Ethernet, то есть «эфирная сеть», отразившее связь новой технологии с ее предшественницей.

Ethernet на коаксиальном кабеле

Подключение к кабелю и передача битов

Пример проводной сети Ethernet на коаксиальном кабеле показан на рис. 2.1, а схематичное устройство коаксиального кабеля — на рис. 2.2.

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 является экраном, защищающим внутреннюю жилу от внешних электромагнитных полей.

Кабель используется как моноканал для всех станций. Приемник и передатчик каждого сетевого адаптера соединяются с медным проводником кабеля, так что электрические сигналы, посланные каким-либо передатчиком, распространяются по кабелю и *принимаются всеми приемниками* — эффект разделяемой среды налицо.



Рис. 2.1. Сеть Ethernet на коаксиальном кабеле

В каждый момент времени должен работать только один передатчик, иначе электрические сигналы, посланные нескольким передатчиками, будут накладываться друг на друга и искажаться. Такая ситуация называется **коллизией**; она нежелательна, но возможна, далее мы увидим, каким образом сетевые адAPTERы Ethernet реагируют на коллизии.

Для того чтобы электрические сигналы не отражались от концов кабеля и не создавали помех даже при работе одного передатчика, на концах кабеля обязательно устанавливаются **терминальные резисторы**, которые ослабляют дошедшие до них сигналы.

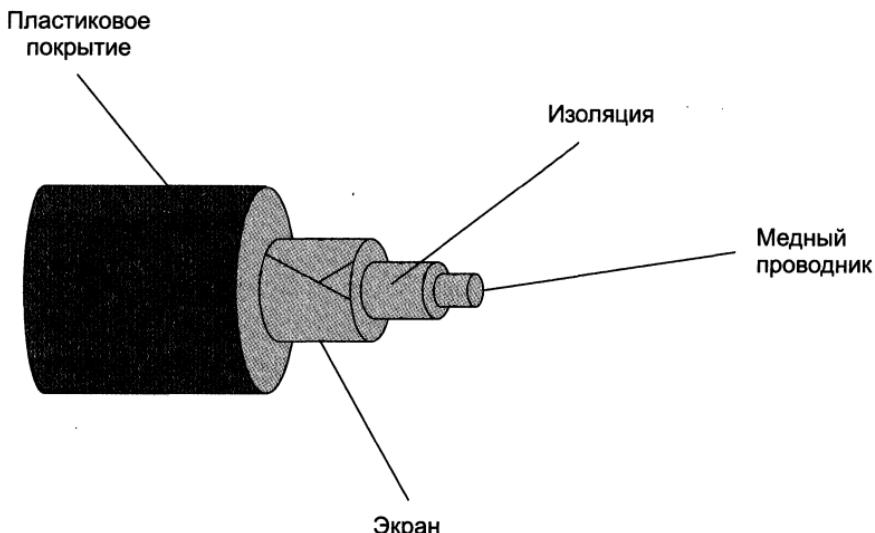


Рис. 2.2. Устройство коаксиального кабеля

Сетевые адаптеры коаксиального варианта Ethernet передают данные со скоростью 10 Мбит/с. Для устойчивой передачи двоичные данные кодируются так называемым **манчестерским кодом**, когда двоичные единицы представляются перепадом электрического сигнала от низкого потенциала к высокому, а двоичные нули — обратным перепадом (рис. 2.3).

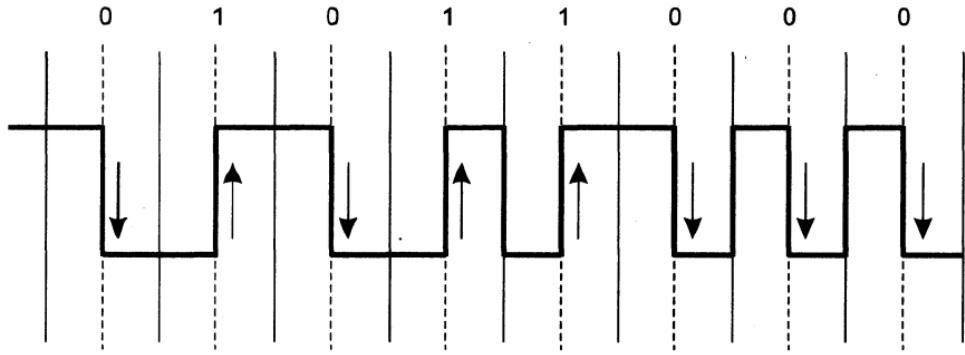


Рис. 2.3. Манчестерский код

Манчестерский код обладает одним важным для локальных сетей свойством — он обеспечивает надежную синхронизацию приемника по отношению к передатчику.

Синхронизация передатчика и приемника нужна для того, чтобы приемник точно знал, в какой момент времени необходимо считывать новую порцию информации с линии связи. Проблема синхронизации в сетях решается сложнее, чем при обмене данными между близко расположенным устройствами, например, между блоками внутри компьютера или же между компьютером и принтером. На небольших расстояниях хорошо работает основанная на отдельной тактирующей линии связи схема, в которой 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 значительно позже или раньше соответствующего сигнала данных, из-за чего бит данных будет пропущен или считан повторно. Другой причиной, по которой в сетях отказываются от применения тактирующих импульсов, является экономия проводников в дорогостоящих кабелях.

При использовании манчестерского кода на каждом такте происходит по крайней мере одно изменение сигнала на входе приемника (при последовательности единиц или нулей — два), так что приемнику легче «подсинхронизироваться» под темп работы передатчика.

В целом сеть Ethernet на коаксиале представляет собой распределенную по зданию электрическую схему, все элементы которой влияют друг на друга, поэтому для того, чтобы добиться устойчивой работы, нужно соблюдать

довольно много ограничений: на длину сегмента коаксиального кабеля, на его электрические характеристики, на характеристики передатчиков и приемников и т. п.

Адресация и кадры Ethernet

Мы рассмотрели, как по коаксиальному кабелю передаются биты, теперь нам нужно подняться на уровень выше и изучить механизм передачи *кадров* Ethernet.

Ethernet является *дейтаграммным* протоколом, в его функции входит доставка кадра от узла отправителя к узлу назначения без предварительного установления соединения и без повторной доставки искаженных или утерянных кадров.

Кадры Ethernet состоят из заголовка фиксированной длины и структуры и переменного поля данных (рис. 2.4).

6	6	2	46 - 1500	4
DA	SA	T	Data	FCS

Рис. 2.4. Формат кадра Ethernet

Первые два поля заголовка отведены под адреса:

- SA (Source Address) – адрес узла назначения;
- DA (Destination Address) – адрес узла отправителя.

Вообще-то 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, Ethernet же только выполнит такое действие, если с сетевого уровня поступит соответствующее указание.

Для идентификации узлов в Ethernet используются так называемые MAC-адреса. **MAC** является аббревиатурой от названия Media Access Control (управление доступом к среде). **Управление доступом к среде** – одна из основных процедур в сетевых технологиях, использующих разделяемую среду.

В Ethernet управление доступом осуществляется в соответствии с алгоритмом, получившим название **коллективного доступа с опознаванием несущей и обнаружением коллизий** (Carrier Sense Multiple Access with Collision Detection, CSMA/CD). Этот метод стандартизован комитетом IEEE 802.3 и иногда используется как синоним названия Ethernet.

Мы рассмотрим метод CSMA/CD немного позже, а пока остановимся на формате MAC-адресов и процедуре передачи кадров между узлами при отсутствии коллизий.

MAC-адрес состоит из 6 байт, которые обычно записывают в виде шести пар шестнадцатеричных цифр, разделенных дефисом или двоеточиями, например 11-A0-17-3D-BC-01.

Каждый сетевой адаптер имеет по крайней мере один MAC-адрес, который должен его уникально определять в пределах сети.

Протокол Ethernet получает от верхнего уровня данные, которые нужно передать, а также значение MAC-адреса назначения. На основе этой информации протокол Ethernet (а точнее, его реализация в виде драйвера сетевого адаптера и аппаратуры сетевого адаптера) создает новый кадр, помещает адрес и данные в соответствующие поля. Размер данных не должен превышать 1500 байт, об этом заботится протокол верхнего уровня. В поле MAC-адреса источника помещается собственный адрес сетевого адаптера, а в поле T (Type, или EtherType) — условный код протокола верхнего уровня, данные которого находятся в поле данных кадра, например, шестнадцатеричное значение 08-00 соответствует протоколу IP. Так, если в сети, приведенной на рис. 2.1, компьютер с адресом MAC 1 отправляет кадр компьютеру с адресом MAC 4, то в этом кадре в поле адреса назначения DA будет помещено значение MAC 4, а в поле адреса источника SA — значение MAC 1.

Последним этапом формирования кадра является вычисление **контрольной последовательности кадра** (суммы) и помещение ее в поле FCS (Frame Check Sequence). Контрольная последовательность вычисляется по некоторому алгоритму как функция от всех байтов кадра. По значению FCS узел назначения после приема кадра сможет определить, были или не были искажены данные кадра в процессе передачи по сети.

MAC-адрес может определять как индивидуальный интерфейс (как в рассмотренном примере), так и группу интерфейсов или даже все интерфейсы сети. Первый (младший) бит старшего байта адреса назначения является признаком того, является ли адрес индивидуальным или групповым. Если он равен 0, то адрес является **индивидуальным**, то есть идентифицирует один сетевой интерфейс, а если он равен 1, то **групповым**. 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, то есть имеет шестнадцатеричное представление 0xFFFFFFFFFFFF, он идентифицирует все узлы сети и называется **широковещательным**. Кадр с широковещательным адресом назначения должен быть принят всеми узлами сети.

Второй бит старшего байта адреса определяет способ назначения адреса — **централизованный** или **локальный**. Если этот бит равен 0, то адрес назначен

централизовано комитетом IEEE, если же он равен 1, адрес назначен локально администратором сети, в обязанности которого входит обеспечение уникальности локальных адресов, используемых сетью.

Уникальность централизованно назначаемых адресов обеспечивается за счет того, что каждый производитель оборудования получает от IEEE так называемые **организационно уникальные идентификаторы** (Organizationally Unique Identifier, OUI). Каждый производитель помещает выделенный ему идентификатор в 3 старших байта адреса (например, идентификатор 0x0020AF определяет компании ЗСОМ, а 0x00000C – компании Cisco), а уникальность младших трех байтов адреса производитель обеспечивает сам. Уникальные MAC-адреса обычно прожигаются в памяти сетевых адаптеров и интерфейсов Ethernet, поэтому они иногда называются аппаратными адресами. Двадцать четыре бита, отводимые производителю для адресации интерфейсов его продукции, позволяют выпустить примерно 16 миллионов интерфейсов под одним идентификатором организации.

Метод доступа CSMA/CD

После того как кадр сформирован, он может передаваться сетевым адаптером в разделяемую среду.

Перед тем как начать передачу кадра в сеть, сетевой адаптер должен проверить, свободна ли разделяемая среда. Это достигается «прослушиванием» среды, а именно определением наличия несущего сигнала с частотой 10 МГц. Если такого сигнала нет, среда считается свободной, и передатчик может начать передачу в среду своих данных.

Если разделяемая среда *свободна*, то передатчик сетевого адаптера начинает последовательно передавать биты кадра, которые распространяются по коаксиальному кабелю и доходят до каждого подключенного к нему приемника. При передаче кадра в сеть он всегда предваряется **преамбулой** из 8 байт, при этом первые 7 байт состоят из двоичных кодов 10101010, а 8-й байт – это код 10101011. Преамбула нужна для синхронизации приемника с передатчиком.

Все приемники копируют приходящие биты во внутренний буфер своего сетевого адаптера. После этого сетевой адаптер выполняет проверку MAC-адреса назначения, и если он не совпадает с собственным, то кадр из буфера удаляется. Если же адрес показывает, что кадр предназначен данному сетевому адаптеру, то выполняется подсчет контрольной суммы полученных байтов кадра и сравнение полученной суммы со значением поля FCS в кадре. Совпадение этих значений говорит о том, что с большой степенью вероятности кадр был передан сетью без искажений и его поле данных можно передать протоколу верхнего уровня для последующей обработки. В противном случае фиксируется ошибка, кадр из сетевого адаптера удаляется и на обработку «наверх» не передается. Так как протокол Ethernet не занимается восстановлением искаженных кадров, то на этом его функции по обработке пришедшего кадра заканчиваются.

Если же «прослушивание» среды показало, что она занята, то сетевой адаптер продолжает «слушать» среду, дожидаясь момента ее освобождения, и только потом начинает передачу кадра.

Метод доступа Ethernet оговаривает важный временной параметр — обязательную паузу в 9,6 мкс между двумя кадрами одного и того же узла. Эта пауза предотвращает монополизацию разделяемой среды одним узлом; без такой обязательной паузы один узел, захватив среду, мог бы ее использовать неопределенно долгий период времени, если бы верхние уровни постоянно снабжали его информацией для передачи в сеть.

Метод прослушивания среды и пауза между кадрами не гарантируют исключения такой ситуации, когда два или более сетевых адаптера решают, что сеть свободна, и начинают одновременно передавать свои кадры, при этом содержимое обоих кадров сталкивается на общем кабеле, что ведет к искажению информации. Ранее мы уже говорили, что такая ситуация называется **коллизией**. В технологии Ethernet для разрешения коллизий применяется метод доступа CSMA/CD.

ВНИМАНИЕ

Ситуация, когда один передатчик, прослушивая линию, обнаруживает ее занятой, не является коллизией, коллизия возникает лишь при одновременной передаче из нескольких источников.

Рисунок 2.5 иллюстрирует возникновение коллизии. Узел 1, обнаружив, что среда свободна, начал передачу кадра. Через некоторое время после начала передачи кадра узлом 1 узел 2 пытается начать передачу своего кадра. Для этого он прослушивает среду и обнаруживает, что она свободна, так как из-за конечной скорости распространения электрических сигналов по кабелю сигналы передатчика узла 1 просто не успели дойти по кабелю до точки подключения приемника 2.

Некоторое время сетевые адаптеры узлов 1 и 2 передают биты своих кадров одновременно, не замечая коллизии (рис. 2.6). Однако в какой-то момент (момент T_2) приемники всех узлов сети обнаруживают, что сигналы в разделяемой среде искажаются (существует несложный способ обнаружения такого состояния среды) и фиксируют состояние коллизии. При этом все передатчики, которые передают данные в среду, должны прекратить передачу — в нашем примере это передатчики узлов 1 и 2.

После этого передатчики, прервавшие передачу кадров из-за коллизии, должны сделать паузу и попытаться начать передачу своих кадров снова, если к этому моменту среда окажется свободной.

Длительность паузы у каждого передатчика выбирается случайным образом из диапазона от 0 до 52 мкс.

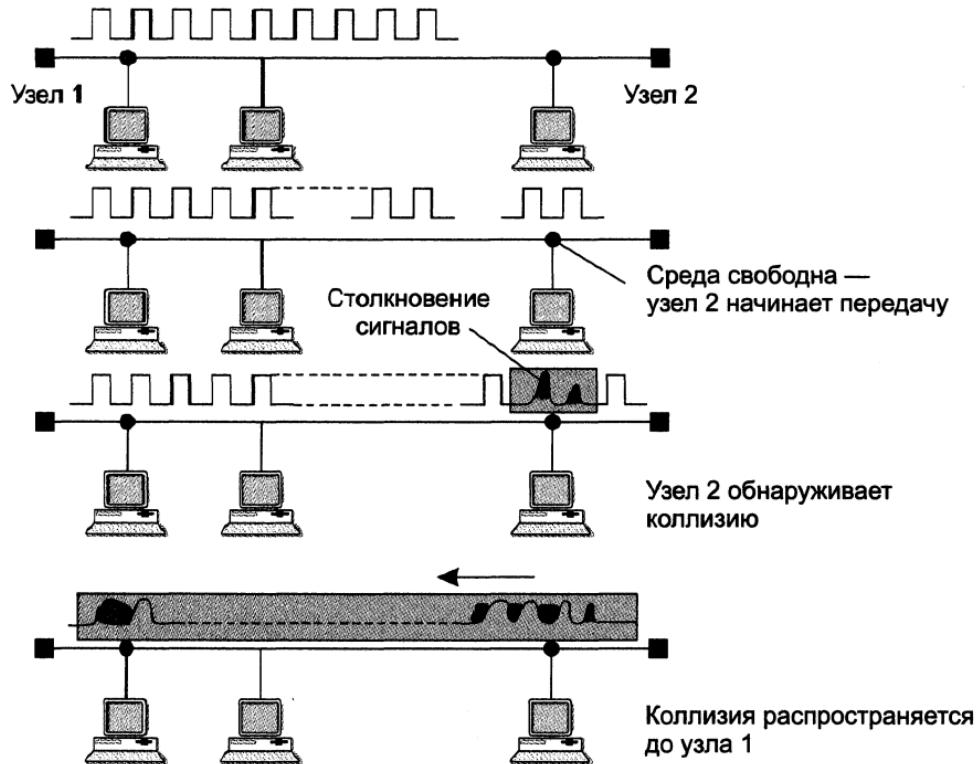


Рис. 2.5. Возникновение и распространение коллизии

Такой прием позволяет разнести по времени моменты попыток начала передачи участвовавшими в коллизии узлами и тем самым снизить вероятность того, что и новая попытка снова приведет к коллизии. В нашем примере узел 1 выбрал паузу продолжительностью p_1 , а узел 2 — p_2 . Так как p_2 меньше, чем p_1 , то узел 2 проводит проверку состояния среды раньше, в момент T_4 , и, обнаружив, что среда свободна, повторно начинает передавать свой кадр F_2 . Пауза узла 1 истекает позже в момент T_5 , так что его проверка состояния среды показывает ее занятость (будем считать, что сигналы от передатчика 2 уже дошли к этому моменту до точки подключения к кабелю приемника 1), и передатчику 2 приходится ждать, пока узел 2 закончит передавать свой кадр F_2 и среда освободится.

Алгоритм доступа к среде предусматривает возможность нескольких последовательных коллизий при попытке передать один и тот же кадр. С каждой новой коллизией интервал, из которого случайным образом выбирается пауза, увеличивается, так что вероятность повторной коллизии снижается.

Описанный алгоритм вычисления последовательных значений паузы носит название **усеченного экспоненциального двоичного алгоритма отсрочки**.

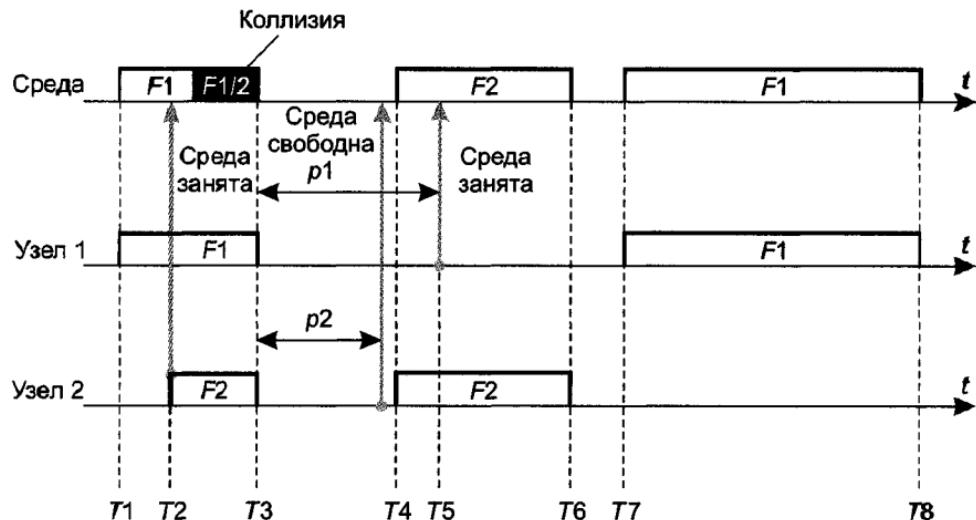


Рис. 2.6. Отработка коллизии

Если 16 последовательных попыток передать кадр приводят к коллизиям, узел должен прекратить попытки передать кадр и просто удалить его из своего буфера, оставив проблему передачи своих данных протоколам верхнего уровня.

«Старательность», с какой алгоритм доступа CSMA/CD пытается передать кадр (до 16 повторных попыток), объясняется просто стремлением повысить производительность сети. Обычно стек протоколов устроен так, что все временные интервалы существенно увеличиваются по мере продвижения вверх от канального к прикладному уровню. Достаточно сравнить паузу, которую делает Ethernet при первом обнаружении коллизии (не более 52 мкс) с тайм-аутом протокола IP (60 с), представляющим собой максимальное время, в течение которого протокол IP ждет прихода очередного фрагмента пакета, после чего пакет считается утерянным. Разница в масштабе в миллион раз впечатляет. Причина, по которой протоколы верхних уровней более инерционны, чем протоколы нижних, объясняется тем, что на верхних уровнях требуется учитывать все возможные случаи, в том числе работу в глобальной составной сети, где пакет может передаваться между конечными узлами довольно долго, например, из-за ожидания в очередях многочисленных промежуточных маршрутизаторов или путешествий по протяженным спутниковым каналам.

В сетях Ethernet на коаксиале расстояния между узлами небольшие, а промежуточных маршрутизаторов или коммутаторов вовсе нет — это позволяет работать с короткими тайм-аутами и быстро организовывать повторную передачу искаженных кадров. Если бы алгоритм доступа протокола Ethernet

оставлял решение задачи повторной передачи искаженного коллизией кадра верхним уровням, то производительность сети была бы гораздо ниже.

Стандарты Ethernet вводят ограничение на *максимальную длину отдельного непрерывного отрезка коаксиального кабеля*. Оно связано с тем, что электрические сигналы передатчиков затухают в процессе прохождения по кабелю, поэтому для того, чтобы не делать слишком мощные передатчики, длину отрезка кабеля ограничивают. Конкретное значение зависит от типа кабеля. В сетях Ethernet исторически использовались два типа коаксиального кабеля: так называемый **«толстый» коаксиал** с внешним диаметром чуть больше сантиметра и **«тонкий» коаксиал** с внешним диаметром около половины сантиметра. Толстый коаксиал обладает лучшими электрическими характеристиками, чем тонкий, поэтому его максимальная длина равна 500 метров, в то время как для тонкого коаксиала это значение составляет 185 метров.

Еще одно ограничение накладывается на так называемый *диаметр сети* — максимальное расстояние между узлами сети. Независимо от типа кабеля диаметр сети Ethernet на разделяемой среде не должен превышать 2500 метров. С учетом ограничения на длину отрезка кабеля, такая протяженность сети может быть достигнута только с применением специальных устройств — повторителей.

Повторитель, как это и следует из его названия, повторяет сигналы, приходящие из одного физического сегмента кабеля в другие сегменты, улучшая их физические характеристики — мощность и форму сигналов, а также синхронность следования (исправляет неравномерность интервалов между импульсами).

Для физического соединения сегментов кабеля с целью увеличения общей длины сети используются **двухпортовые повторители**.

Ethernet на витой паре

Одним из существенных недостатков Ethernet на коаксиальном кабеле является отсутствие оперативной информации о состоянии кабеля и сложность 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 годов, 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, так как

многие здания уже были оснащены нужной кабельной системой. Оставалось разработать способ подключения сетевых адаптеров и прочего коммуникационного оборудования к витой паре таким образом, чтобы изменения в сетевых адаптерах и программном обеспечении сетевых операционных систем были минимальными 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.

Правда, для соединения узлов в сеть теперь обязательно требуется коммуникационное устройство — **многопортовый повторитель** Ethernet на витой паре.

Устройство такого повторителя схематично изображено на рис. 2.7. Каждый сетевой адаптер соединяется с повторителем двумя витыми парами. Одна витая пара 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, кроме того, с которого поступили сигналы, одновременно улучшая их электрические характеристики.

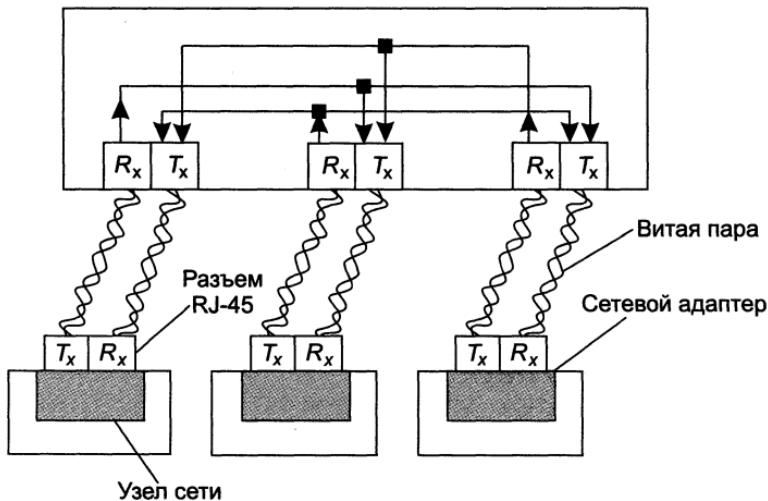


Рис. 2.7. Повторитель Ethernet на витой паре

Многопортовый повторитель часто называют **концентратором**, или **хабом** (от английского 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 иерархическим способом, образуя *древовидную структуру* (рис. 2.8). Добавление каждого хаба изменяет физическую структуру, но оставляет без изменения логическую структуру сети. То есть независимо от количества хабов в сети сохраняется одна общая для всех интерфейсов разделяемая среда, так что передача кадра с любого интерфейса блокирует передатчики всех остальных интерфейсов.

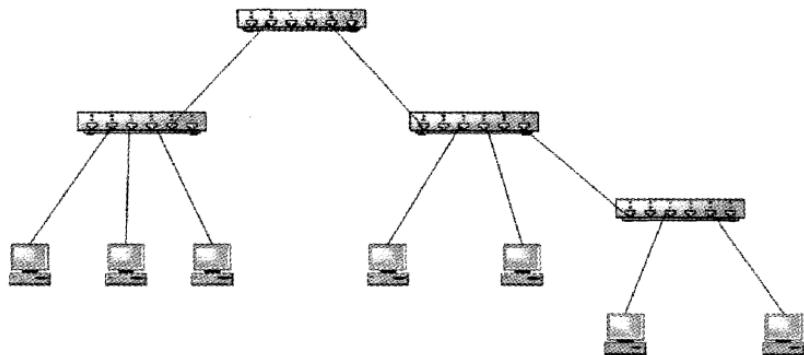


Рис. 2.8. Иерархическое соединение хабов

Физическая структуризация сетей, построенных на основе витой пары, повышает надежность и упрощает обслуживание сети, поскольку в этом случае появляется возможность контролировать состояние и локализовывать 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.

Сети Token Ring и FDDI

Token Ring и FDDI – это функционально намного более сложные технологии, чем Ethernet на разделяемой среде. Разработчики этих технологий стремились наделить сеть на разделяемой среде многими положительными качествами: сделать механизм разделения среды предсказуемым и управляемым, обеспечить отказоустойчивость сети, организовать приоритетное обслуживание для чувствительного к задержкам трафика, например голосового. Нужно отдать им должное – во многом их усилия оправдались, и сети FDDI довольно долгое время успешно использовались как магистрали сетей масштаба кампуса, в особенности в тех случаях, когда нужно было обеспечить высокую надежность магистрали.

Механизм доступа к среде в сетях Token Ring и FDDI является более детерминированным, чем в сетях Ethernet.

Рассмотрим его на примере сети **Token Ring**, станции которой связаны в кольцо (рис. 2.9), так что любая станция непосредственно получает данные

только от одной станции — той, которая является предыдущей в кольце, а передает данные своему ближайшему соседу вниз по потоку данных. Скорость передачи данных в первых сетях Token Ring, разработанных компанией IBM, была всего 4 Мбит/с, но затем была повышена до 16 Мбит/с. Основная среда передачи данных — витая пара. Для адресации станций сети Token Ring (и FDDI) используют MAC-адреса того же формата, что и Ethernet.

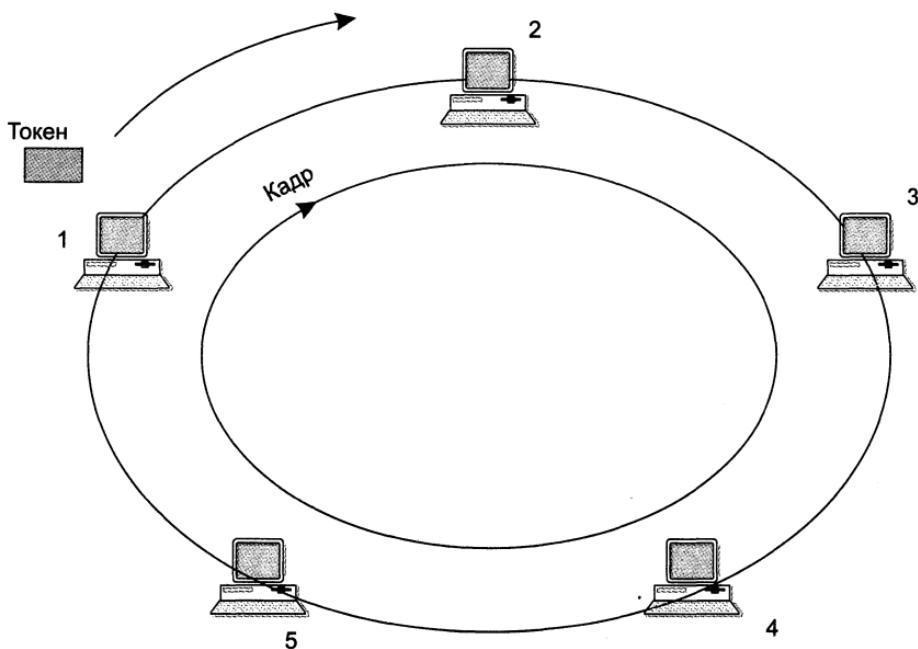


Рис. 2.9. Сеть Token Ring

Метод доступа Token Ring основан на передаче специального кадра — **токена доступа** — от узла к узлу, при этом только узел, владеющий токеном, может передавать свои кадры в кольцо, которое становится в этом случае разделяемой средой. Существует лимит на период монопольного использования среды — это так называемое **время удержания токена**, по истечении которого станция обязана передать токен своему соседу по кольцу. В результате такие ситуации, как неопределенное время ожидания доступа к среде, характерные для Ethernet, здесь исключены (по крайней мере, в тех случаях, когда сетевые адAPTERЫ станций исправны и работают без сбоев). Максимальное время ожидания всегда нетрудно оценить, так как оно равно произведению времени удержания токена на количество станций в кольце. Так как станция, получившая токен, но не имеющая в этот момент кадров для передачи, передает токен следующей станции, время ожидания может быть меньше.

Отказоустойчивость сети Token Ring определяется использованием в сети повторителей (не показанных на рис. 2.9) для создания кольца. Каждый такой повторитель имеет несколько портов, которые образуют кольцо за счет внутренних связей между передатчиками и приемниками. В случае отказа

или отсоединения станции повторитель организует обход порта этой станции, так что связность кольца не нарушается.

Поддержка чувствительного к задержкам трафика достигается за счет *системы приоритетов кадров*. Решение о приоритете конкретного кадра принимает передающая станция. Токен также всегда имеет некоторый уровень текущего приоритета. Станция имеет право захватить переданный ей токен только в том случае, если приоритет кадра, который она хочет передать, выше приоритета токена (или равен ему). В противном случае станция обязана передать токен следующей по кольцу станции.

Благодаря более высокой, чем в сетях Ethernet, скорости, детерминированности распределения пропускной способности сети между узлами, а также лучших эксплуатационных характеристик (обнаружение и изоляция неисправностей), сети Token Ring были предпочтительным выбором для таких чувствительных к подобным показателям приложений, как банковские системы и системы управления предприятием.

Технологию FDDI можно считать усовершенствованным вариантом Token Ring, так как в ней также используется метод доступа к среде, основанный на передаче токена, а также кольцевая топология связей, но вместе с тем FDDI работает на более высокой скорости и имеет более совершенный механизм отказоустойчивости.

Технология FDDI стала первой технологией локальных сетей, в которой использовалось оптическое волокно, начавшее применяться в телекоммуникационных сетях с 70-х годов прошлого века в качестве разделяемой среды передачи данных. За счет применения оптических систем скорость передачи данных удалось повысить до 100 Мбит/с (позже появилось оборудование FDDI на витой паре, работающее на той же скорости).

В тех случаях, когда нужно было обеспечить высокую надежность сети FDDI, применялось двойное кольцо (рис. 2.10). В нормальном режиме станции используют для передачи данных и токена доступа первичное кольцо, а вторичное пристаивает¹. В случае отказа, например, при обрыве кабеля между станциями 1 и 2, как показано на рис. 2.10, первичное кольцо объединяется с вторичным, вновь образуя единое кольцо. Этот режим работы сети называется **режимом свертывания колец**. Операция свертывания производится средствами повторителей (не показанных на рисунке) и (или) сетевых адаптеров FDDI. Для упрощения этой процедуры данные по первичному кольцу всегда передаются в одном направлении (на диаграммах это направление изображается против часовой стрелки), а по вторичному — в обратном (изображается по часовой стрелке). Поэтому при формировании общего кольца из двух колец передатчики станций по-прежнему остаются подключенными

¹ Существовали фирменные реализации оборудования FDDI, в которых в нормальном режиме использовалось и вторичное кольцо. Тем самым удавалось добиваться удвоения скорости передачи данных.

к приемникам соседних станций, что позволяет правильно передавать и принимать информацию соседними станциями.

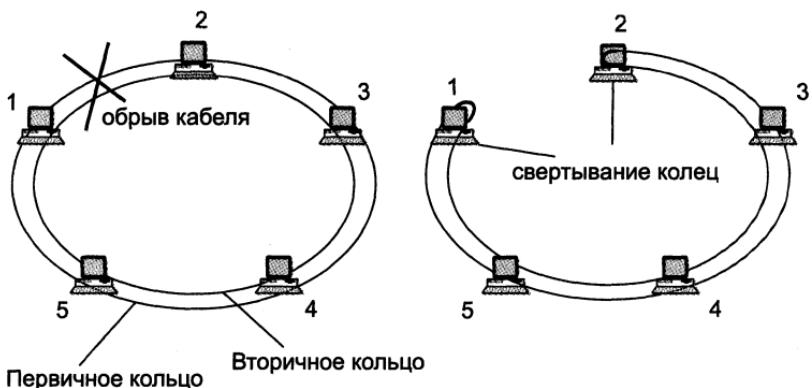


Рис. 2.10. Отказоустойчивость в сети FDDI

В стандартах FDDI много внимания отводится различным процедурам, которые позволяют определить наличие отказа в сети, а затем произвести необходимое реконфигурирование. Технология FDDI расширяет механизмы обнаружения отказов технологии Token Ring за счет резервных связей, которые предоставляет второе кольцо.

Достоинства и недостатки разделяемой среды

Использование стандартных технологий локальных сетей на разделяемой среде имеет ряд достоинств:

- простая топология сети допускает легкое наращивание количества узлов (в небольших пределах);
- отсутствуют потери кадров из-за переполнения буферов коммуникационных устройств, так как сам метод доступа к разделяемой среде регулирует поток кадров и приостанавливает станции, слишком часто генерирующие кадры;
- простота протоколов обеспечивает низкую стоимость сетевых адаптеров, повторителей и концентраторов и сети в целом.

Однако сам принцип разделения среды предопределяет и главный недостаток этого класса технологий — неспособность преодолеть дефицит пропускной способности в случае его появления.

Такой дефицит может возникнуть, например, при увеличении узлов сети, когда постоянная пропускная способность среды начинает делиться между большим количеством узлов, а значит, доля, приходящаяся на отдельный узел,

уменьшается. Появление в компьютерах сети приложений, например, мультимедийных, пересылающих по сети больше данных, также ведет к дефициту пропускной способности.

Усугубляет ситуацию случайный характер запросов к разделяемой среде.

Результатом неупорядоченности во времени запросов к среде является резкий, близкий к экспоненциальному, рост времени ожидания доступа к среде после превышения некоторого предела нагрузки на среду.

Типичные для сетей Ethernet, Token Ring и FDDI зависимости времени ожидания доступа от ее коэффициента использования ρ показаны на рис. 2.11. Под коэффициентом использования среды понимается отношение предлагаемой нагрузки (то есть суммарная скорость данных, которую узлы сети хотели бы передать по сети) к пропускной способности среды.

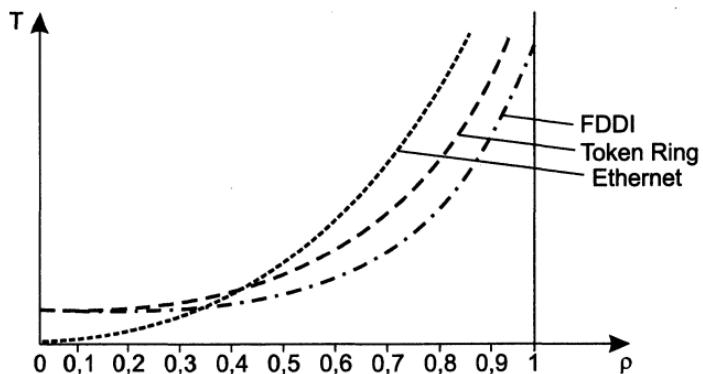


Рис. 2.11. Задержки доступа к среде передачи данных для технологий Ethernet, Token Ring и FDDI

Как видно из рисунка, порог, при котором наступает резкий переход в поведении сети и почти прямолинейная зависимость переходит в крутую экспоненциальную, для разных технологий различен. Хуже всего проявляется себя Ethernet – переход наступает уже при коэффициенте использования сети в 30–50 %, так как здесь сказывается отрицательный эффект коллизий. В сетях Token Ring, для которых характерен более сложный детерминированный метод доступа, этот порог увеличивается до 60 %, а в сетях FDDI он вырастает до 70–80 %, поскольку в FDDI применяется усовершенствованный по сравнению с Token Ring алгоритм доступа.

Итак, проблема дефицита пропускной способности в сетях с разделяемой средой является «врожденной», а повышение скорости передачи данных не может считаться хорошим универсальным решением, так как новые высокоскоростные версии какой-либо технологии, например Ethernet, появляются не каждый день, да и переход на новую скорость требует замены

сетевого оборудования. Помимо проблем с пропускной способностью, разделяемая среда, как указывалось ранее, имеет ограничения на длину отрезков кабеля и диаметр.

Таким образом, требовалось другое, более гибкое решение, обеспечивающее расширение и развитие локальной сети без перехода на новую технологию. Такое решение было найдено в виде коммутаторов локальных сетей, которые сначала применялись как мосты между несколькими сегментами разделяемой среды, а потом полностью вытеснили разделяемую среду в сфере проводных локальных сетей.

Коммутируемые сети Ethernet

Принцип работы коммутаторов Ethernet

Проблемы, возникающие из-за наличия единой разделяемой среды, были распознаны разработчиками и пользователями локальных сетей достаточно быстро, и ответом стало появление устройства под названием «мост».

Мост локальной сети (LAN bridge) выполняет логическую структуризацию сети, разделяя единую разделяемую среду на несколько сегментов.

Мост передает кадры с одного своего порта на другой, анализируя MAC-адреса назначения, помещенные в эти кадры. Передача происходит только в том случае, когда действительно необходимо передать кадр из одного сегмента в другой. В том же случае, когда источник и приемник находятся в одном сегменте, мост игнорирует кадр.

Как видно из этого краткого описания, применение моста приводит к снижению нагрузки на разделяемую среду, так как каждый сегмент сети передает только свой внутрисегментный трафик плюс межсегментный, который поступает из другого сегмента или направляется в другой сегмент.

Другими словами, мост *делит* одну разделяемую среду на несколько разделяемых сред по числу подключенных к нему сегментов. На рис. 2.12 показана сеть, образованная мостом, к которому подключены два сегмента, причем каждый сегмент располагает собственной разделяемой средой. Если бы в этой сети на месте моста был установлен хаб, то сеть представляла собой одну разделяемую среду и на каждый компьютер приходилась бы $1/6$ общей пропускной способности сети. Использование же моста повышает эту долю в предельном случае (когда межсегментный трафик пренебрежимо мал) до $1/3$.

Помимо снижения нагрузки применение мостов позволяет увеличить диаметр сети.

Алгоритм работы моста был стандартизован IEEE в 1990 году в стандарте 802.1D. Этот алгоритм получил название **алгоритма прозрачного моста**, так как наличие такого моста никак не сказывается на работе сетевых адаптеров компьютеров, то есть мост прозрачен для них.

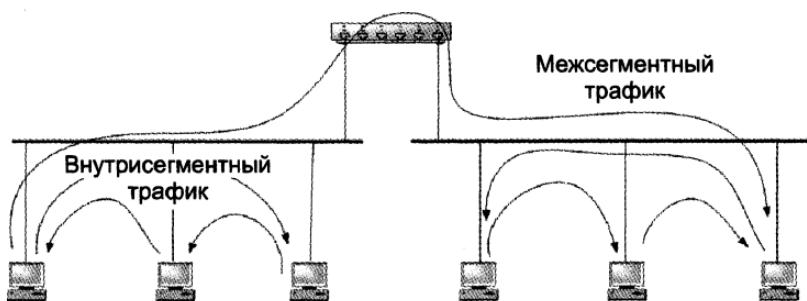


Рис. 2.12. Сегментация локальной сети

Алгоритм прозрачного моста не зависит от технологии локальной сети, в которой устанавливается мост (коммутатор), поэтому прозрачные мосты (коммутаторы) Ethernet работают точно так же, как прозрачные мосты (коммутаторы) FDDI или Token Ring.

Достаточно долго в локальных сетях применялись мосты с небольшим количеством портов, так как разделяемая среда по-прежнему широко использовалась как основной строительный элемент локальной сети. Мости были медленными и дорогими. Ситуация изменилась в начале 90-х годов, когда появились так называемые коммутаторы локальных сетей.

Коммутатор локальной сети (LAN switch) функционально подобен мосту — он работает по тому же алгоритму. Коммутаторы отличаются от традиционных мостов потребительскими характеристиками: большим количеством портов, высокой скоростью передачи кадров с порта на порт, низкой стоимостью.

Такое сочетание характеристик привело к тому, что стал возможным *полный отказ от разделяемой среды*. Этот отказ происходил постепенно, и к настоящему времени его можно считать свершившимся — производство повторителей прекращено и купить их сейчас можно только «с рук» или из старых запасов мелких поставщиков.

Рассмотрим алгоритм работы коммутатора на примере сети Ethernet, показанной на рис. 2.13. Для продвижения кадров коммутатор использует **таблицу продвижения**, которая состоит из записей, включающих два поля:

- MAC-адрес назначения;
- идентификатор порта коммутатора, на который нужно передать кадр с указанным адресом назначения.

В исходном состоянии (которое и показано на рис. 2.13) таблицы продвижения коммутаторов сети пусты.

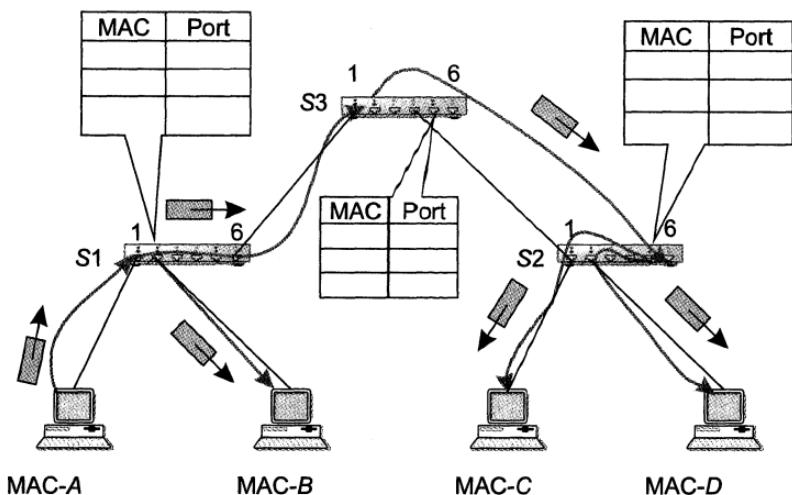


Рис. 2.13. Алгоритм работы коммутатора, начальный этап — адреса не изучены

Коммутатор строит свою адресную таблицу автоматически на основании пассивного наблюдения за трафиком, циркулирующим в подключенных к его портам сегментах. При этом коммутатор учитывает адреса источников кадров данных, поступающих на порты коммутатора. По адресу источника кадра коммутатор делает вывод о принадлежности узла-источника тому или иному порту. Сами порты коммутатора не адресуются при передаче кадров¹, они работают в так называемом неразборчивом режиме захвата кадров, когда все поступающие на порт кадры, независимо от их адреса назначения, запоминаются на время в буферной памяти.

Пусть теперь компьютер *A* посыпает кадр компьютеру *C*. Компьютер *A* соединен кабелем с портом 1 коммутатора *S1*, этот кабель выделен в его полное распоряжение и не разделяется с другими компьютерами. Сетевой адаптер, подключенный к порту коммутатора, работает в **дуплексном режиме**, когда одновременные передача и прием данных не считаются недопустимой ситуацией, как в сетях Ethernet на разделяемой среде.

Когда коммутатор *S1* получает кадр с адресом назначения MAC-*C*, который отсутствует в его таблице продвижения, он просто копирует этот кадр

¹ Порт коммутатора может иметь (и обычно имеет) MAC-адрес, но этот адрес используется не для передачи кадров данных, а для удаленного управления портом по некоторому протоколу управления (например, протоколу SNMP, который рассматривается в главе 5). Коммутатор в целом тоже имеет MAC-адрес, применяемый, в частности, для удаленного управления.

на все свои порты, кроме того, на который он получил данный кадр. Такой режим называется **затоплением сети**. В нашем примере коммутатор $S1$ копирует кадр на порты 2 и 6, так как к остальным портам ничего не подключено. Первое копирование было очевидно безрезультатным, так как кадр попадает компьютеру B , которому он не предназначался (и который отбрасывает этот кадр). А вот копирование на порт 6 имело смысл, так как кадр теперь попадает в коммутатор $S3$, находящийся на пути кадра к компьютеру C . Коммутатор $S3$ также имеет пустую таблицу продвижения, поэтому он передает кадр на единственный отличный от порта 1 порт, у которого физическое подключение активно, то есть на порт 4.

Коммутатор $S2$, получив кадр, копирует его на порты 1 и 2, причем в первом случае это копирование оказывается результативным, так как к этому порту подключен компьютер назначения C .

Одновременно с затоплением сети коммутаторы строят свои таблицы продвижения. Таблица строится на основании адресов источника кадров, проходящих через коммутатор. Так, коммутатор $S1$, передавая кадр на порты 2 и 6, запомнил тот факт, что кадр от компьютера с адресом MAC-А пришел на порт 1. Поэтому он помещает в свою пустую таблицу продвижения первую запись:

Адрес	Порт
MAC-А	1

Теперь, имея такую таблицу, он начнет обрабатывать кадр, пришедший для компьютера A от любого компьютера сети, более рационально: он не будет копировать его на все порты, а передаст единственную копию на порт 1. Именно так он поступит с кадром, который компьютер C направит в ответ на кадр, полученный им от компьютера A . Ответ от компьютера C позволит коммутатору $S1$ добавить новую запись, в результате чего таблица приобретет следующий вид:

Адрес	Порт
MAC-А	1
MAC-С	6

После этого кадры, посланные для компьютера C , также начинают продвигаться коммутатором $S1$ рационально: единственная копия передается только на порт 6.

После многократных обменов кадрами все коммутаторы помещают MAC-адреса компьютеров сети в свои таблицы, так что они приобретают вид, показанный на рис. 2.14. Теперь кадр, посланный компьютером A компьютеру C уже

не затапливает сеть, а передается в виде единственной копии по кратчайшему пути.

Как видно из описания, коммутаторы Ethernet действительно прозрачны для компьютеров, кроме того, их способность самообучаться без какого бы то ни было предварительного ручного конфигурирования удобна для администратора сети.

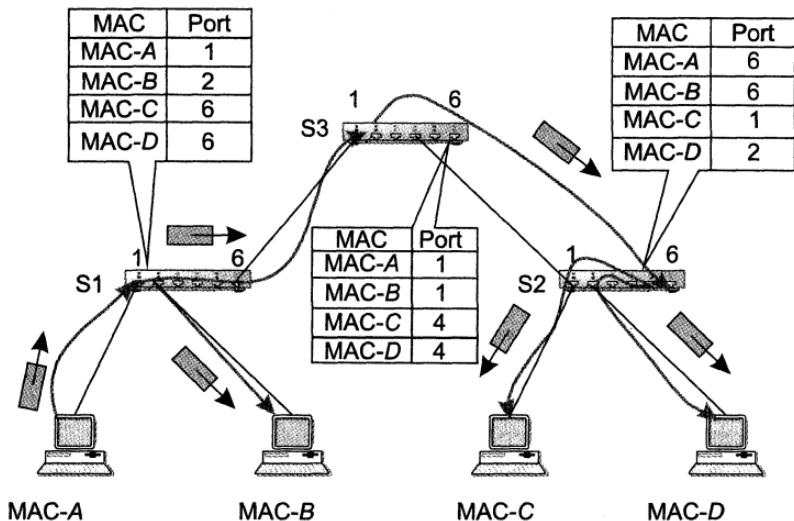


Рис. 2.14. Алгоритм работы коммутатора, конечный этап — все адреса изучены

Коммутируемая версия Ethernet явилась большим шагом вперед в отношении производительности сети по сравнению с Ethernet на разделяемой среде, так как она исключила периоды ожидания освобождения среды за счет параллельного продвижения кадров от всех компьютеров сети.

Высокие скорости работы коммутаторов при обработке кадров и возросшие с 10 Мбит/с до 100 Мбит/с, 1000 Мбит/с и 10 Гбит/с скорости передачи битов по каналам связи также привели к тому, что локальные сети стали очень эффективной и сравнительно простой средой взаимодействия компьютеров.

Однако простота организации коммутируемых сетей Ethernet сопряжена с рядом недостатков.

- Появление кадров с неизвестными ранее адресами приводит к так называемым **широковещательным штормам**, так как эти кадры затапливают сеть своими копиями. Сеть Ethernet не может эффективно предотвращать ситуации, когда какой-либо из компьютеров начинает работать некорректно, генерируя кадры с ошибочным адресом. Говорят, что сеть, построенная на коммутаторах Ethernet, является «плоской», имея в виду

тот факт, что в сети нет естественных барьеров на пути распространения ошибочного широковещательного трафика.

- Плоские адреса Ethernet не очень удобны для адресации узлов больших сетей, так как в этом случае таблицы продвижения содержат слишком много записей. В этом отношении, как уже было отмечено, гораздо эффективнее многоуровневый адрес сетевого уровня.
- Самообучение коммутаторов Ethernet на основе наблюдения за проходящим трафиком приводит к тому, что коммутируемые сети Ethernet эффективно работают только при древовидной топологии сети, когда в сети нет петель. Иначе кадры начинают «зацикливаться» и размножаться, а таблицы продвижения не достигают устойчивого состояния, постоянно перестраиваясь, что, конечно, не дает сети нормально работать. Решить проблему призван протокол покрывающего дерева, позволяющий локальной сети иметь произвольную топологию связей.

Протокол покрывающего дерева

Протокол покрывающего дерева (Spanning Tree Protocol, STP)¹ предназначен для повышения надежности локальной сети на коммутаторах за счет использования произвольной топологии, обеспечивающей альтернативные пути между узлами сети.

Пример такой топологии приведен на рис. 2.15. В этой сети существуют альтернативные пути. Например, между коммутаторами $S1$ и $S3$ есть три пути:

- непосредственная связь между портами 2 каждого из этих коммутаторов;
- путь « $S1$ порт 4» — « $S4$ порт 3» — « $S4$ порт 1» — « $S3$ порт 6»;
- путь « $S1$ порт 4» — « $S4$ порт 3» — « $S4$ порт 4» — « $S5$ порт 5» — « $S5$ порт 2» — « $S3$ порт 3».

Альтернативные пути можно использовать в разных целях, например, для повышения производительности сети, когда трафик к определенному узлу назначения распараллеливается между такими путями, а также для повышения надежности сети. В последнем случае некоторые линии связи между коммутаторами переводятся в резервное состояние, то есть такое, в котором они не применяются для передачи трафика, но могут быть оперативно переведены в рабочее состояние при выходе из строя какого-либо элемента сети — коммутатора, одного из портов или линии связи между портами. Резервные линии связи называют также избыточными.

Протокол STP позволяет автоматически определить, какие линии связи нужно перевести в резервное состояние, чтобы оставшаяся топология была

¹ Часто используется также название «алгоритм покрывающего дерева» (Spanning Tree Algorithm, STA).

древовидной и, следовательно, пригодной для работы коммутаторов. Одна из возможных древовидных топологий выделена на рис. 2.15 жирными линиями. При отказах элементов сети протокол STP заново находит топологию покрывающего дерева, если это возможно. Например, если в нашем примере произойдет отказ коммутатора S_3 , то связь « S_1 порт 4» – « S_4 порт 3» будет автоматически переведена в рабочее состояние и работоспособность сети восстановится.

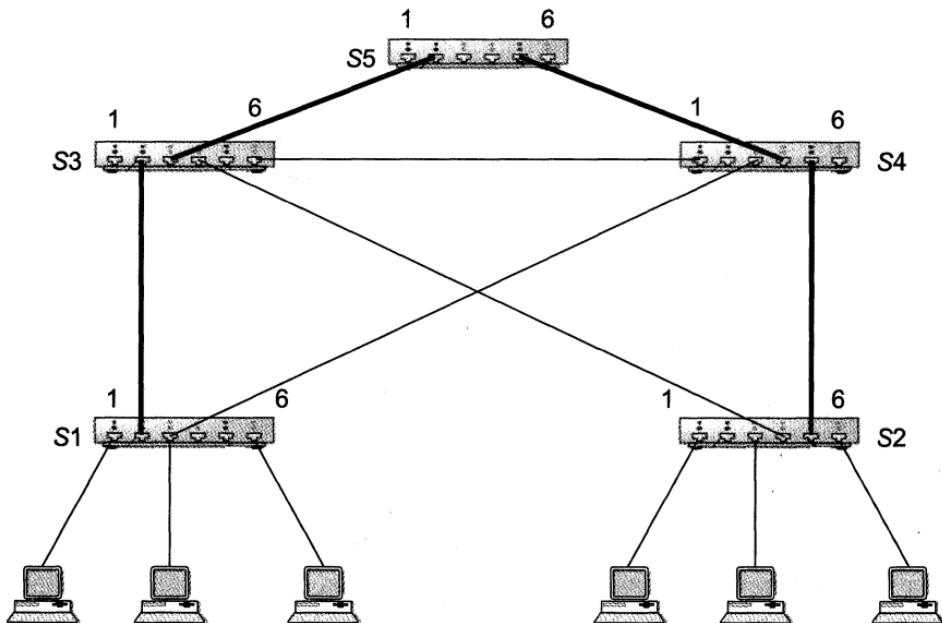


Рис. 2.15. Локальная сеть с альтернативными путями и покрывающее дерево

Работа протокола STP строится на периодическом обмене коммутаторами сети специальными служебными сообщениями, называемыми **протокольными блоками данных моста** (Bridge Protocol Data Unit, **BPDU**). Для доставки BPDU используется групповой адрес 01:80:C2:00:00:00, позволяющий организовать эффективный обмен данными.

Первым этапом построения дерева является выбор его корня — так называемого **корневого коммутатора**. Корневым выбирается коммутатор с меньшим идентификатором, в качестве которого выступает число, полученное конкатенацией приоритета и MAC-адреса коммутатора. Администратор сети может влиять на выбор корневого коммутатора, изменяя приоритеты коммутаторов сети.

После выбора корневого коммутатора у каждого коммутатора (кроме корневого) выбирается **корневой порт** — тот порт, расстояние от которого до корневого коммутатора меньше, чем у других портов коммутатора. Расстояние

(также называемое метрикой) в протоколе STP связано со скоростью линии связи. Для того чтобы найти расстояние от порта одного коммутатора до порта другого коммутатора, нужно просуммировать метрики линий связи, через которые проходит путь от порта до порта. Наличие метрики, отражающей скорость линии связи, характерно для многих протоколов маршрутизации, которые будут изучаться в главе 3. Протокол STP много позаимствовал у этих протоколов, приспособив логику их работы к коммутаторам локальных сетей.

На рис. 2.15 корневым выбран коммутатор S_5 , а корневыми портами — порт 3 коммутатора S_3 , порт 4 коммутатора S_4 , порт 2 коммутатора S_1 и порт 5 коммутатора S_2 . Будем считать, что все линии связи в нашем примере имеют одну и ту же скорость 10 Мбит/с, которой соответствует метрика 10. Из нашего примера видно, что в некоторых случаях выбор корневого порта нельзя выполнить, используя в качестве критерия только расстояние до корневого коммутатора, например, у коммутатора S_1 порты 2 и 4 имеют одинаковое расстояние в 20 единиц до корневого коммутатора S_5 . В таких случаях для устранения неопределенности задействуют приоритеты портов, которые назначаются администратором.

Последним этапом работы протокола STP является выбор **назначенных портов** линий связи. В коммутируемой сети каждая линия связи соединяет два порта, и тот порт, расстояние от которого до корневого коммутатора ближе, становится назначенным портом линии, другой порт линии блокируется, но только в том случае, если он не является корневым. При равных расстояниях выбирается порт с меньшим приоритетом.

Связи, оставшиеся незаблокированными в результате работы данного алгоритма, образуют **граф покрывающего дерева**.

После построения покрывающего дерева коммутаторы начинают передавать через незаблокированные порты кадры данных. В нормальном режиме корневой коммутатор продолжает генерировать кадры BPDU, а остальные коммутаторы получают их через свои корневые порты и ретранслируют через назначенные порты. Если по истечении некоторого тайм-аута корневой порт любого коммутатора сети не получает служебный пакет BPDU, то он инициализирует новую процедуру построения покрывающего дерева.

Основным недостатком протокола STP является его инерционность — выявление отказа элемента сети может занимать до 20 секунд, а построение новой конфигурации — еще 30 секунд.

В 1998 году была стандартизована более быстрая версия этого протокола под названием **Rapid STP** (RSTP). В этой версии обнаружение неисправности и построение нового дерева занимает всего несколько секунд. Основные изменения по сравнению с STP коснулись снижения тайм-аутов и одновременного с выбором корневого и назначенных портов выбора их альтернативных вариантов, поэтому при отказе порта или линии связи переход на альтернативный порт происходит очень быстро, так как он не связан с построением

нового дерева. После принятия стандарта RSTP старая версия, то есть STP, больше не рекомендована для реализации.

Несмотря на усовершенствования, протокол RSTP не смог устранить еще один недостаток STP — неоптимальность единственного дерева для некоторых маршрутов. Так, в описываемом примере передача кадров от компьютера *A* компьютеру *D* происходит через коммутаторы *S1*, *S3*, *S4* и *S2*, в то время как структура линий связи позволяет сделать это проще, исключив коммутатор *S3*. Протоколы маршрутизации свободны от этого недостатка.

Скоростные версии Ethernet

Скорость 10 Мбит/с первой стандартной версии Ethernet долгое время удовлетворяла потребности пользователей локальных сетей. Однако в начале 90-х годов начала 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 низкой стоимости оборудования.

Результатом стало появление новых скоростных стандартов Ethernet: Fast Ethernet (скорость 100 Мбит/с), Gigabit Ethernet (1000 Мбит/с, или 1 Гбит/с) и 10G Ethernet (10 Гбит/с). Во время написания этой книги два новых стандарта — 40G Ethernet и 100G Ethernet — находились в стадии разработки, обещая следующее десятикратное превышение верхней границы производительности Ethernet.

Разработчикам новых скоростных стандартов Ethernet удалось сохранить основные черты классической технологии Ethernet и, прежде всего, простой способ обмена кадрами без встроенных в технологию сложных контрольных процедур. Этот фактор оказался решающим в соревновании технологий локальных сетей, так как выбор пользователей всегда склонялся в пользу простого наращивания скорости сети, а не в пользу решений, связанных с более эффективным расходованием той же самой пропускной способности с помощью более сложной и дорогой технологии. Примером такого подхода служит переход с оборудования Fast Ethernet на Gigabit Ethernet вместо перехода на оборудование ATM 155 Мбит/с. Несмотря на значительную разницу в пропускной способности (1000 Мбит/с против 155 Мбит/с), оба варианта обновления сети примерно равны по степени улучшения «самочувствия» приложений, так как Gigabit Ethernet достигает нужного эффекта за счет равного повышения доли пропускной способности для всех приложений, а ATM перераспределяет меньшую пропускную способность более тонко, дифференцируя ее в соответствии с потребностями приложений.

Тем не менее пользователи предпочли не вдаваться в детали и тонкости настройки сложного оборудования, когда можно просто применить знакомое и простое, но более скоростное оборудование Ethernet.

Большой вклад в «победу» Ethernet внесли и 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 включать вариант работы на разделяемой среде в описание стандарта.

Повышение скорости работы Ethernet было достигнуто за счет улучшения качества кабелей, применяемых в компьютерных сетях, а также совершенствования методов кодирования данных при их передаче по кабелям, то есть за счет совершенствования физического уровня технологии. Именно этим вопросам и будут посвящены несколько следующих разделов данной главы.

Кабели и методы кодирования

Кабели

Мы уже познакомились с двумя типами кабелей, применяемых в локальных сетях: коаксиальным кабелем и кабелем на витой паре. На рис. 2.16 представлен полный спектр кабелей локальных сетей, куда помимо упомянутых кабелей включен также волоконно-оптический кабель.

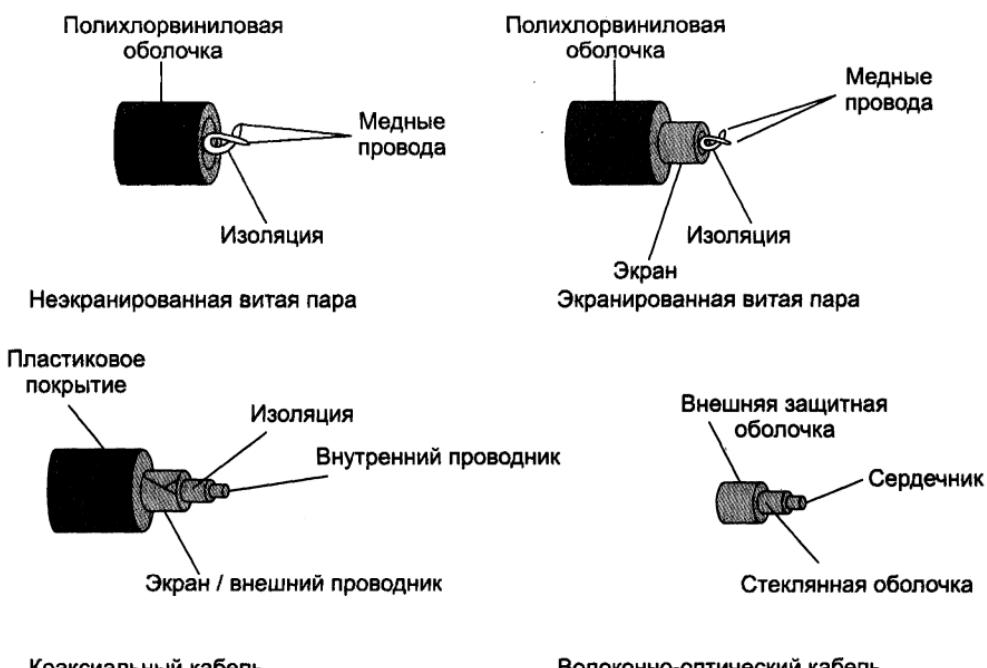


Рис. 2.16. Кабели локальных сетей

Кабели на витой паре наиболее популярны сегодня в локальных сетях, так как являются собой хороший компромисс между стоимостью и характеристиками кабеля, позволяя передавать данные со скоростью до 1 Гбит/с на расстояние до 100 м. Как видно из рисунка, существует два типа кабелей на витой паре: на **экранированной витой паре** (Shielded Twisted Pair, STP) и на **неэкранированной витой паре** (Unshielded Twisted Pair, UTP). Хотя экран, безусловно, повышает защищенность проводников кабеля от внешних помех, неэкранированные кабели UTP гораздо более популярны. Это объясняется тем, что экранированный кабель нужно заземлять, что означает дополнительную работу и дополнительные требования к помещениям. В то же время скручивание проводов само по себе снижает влияние внешних и взаимных помех на полезные сигналы, передаваемые по кабелю, чего оказывается вполне достаточно.

Волоконно-оптический кабель состоит из тонких (5–60 микронов) гибких стеклянных волокон (волоконных световодов), по которым распространяются световые сигналы. Это наиболее качественный тип кабеля — он обеспечивает передачу данных с очень высокой скоростью (до 10 Гбит/с и выше) и к тому же 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 (сердцевины), представляющего собой стеклянное волокно, и стеклянной оболочки, обладающей меньшим показателем преломления, чем сердцевина. Распространяясь по сердцевине, лучи света не выходят за ее пределы, отражаясь от покрывающего слоя оболочки. В зависимости от распределения показателя преломления и величины диаметра сердечника различают многомодовое и одномодовое волокно.

В кабеле на основе **одномодового волокна** (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 (рис. 2.17, а). 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 сложно направить пучок света, не потеряв при этом значительную часть его энергии.

В кабеле на основе **многомодового волокна** (Multi Mode Fiber, MMF) используются более широкие внутренние сердечники, которые легче изготовить технологически. В многомодовых кабелях во внутреннем проводнике одновременно существует несколько световых лучей, отражающихся от внешнего проводника под разными углами (рис. 2.17, б). Угол отражения луча называется **модой** луча. Возникающая при этом интерференция лучей ухудшает качество передаваемого сигнала, что приводит к искажениям передаваемых импульсов в многомодовом оптическом волокне. По этой причине технические характеристики многомодовых кабелей хуже, чем одномодовых.

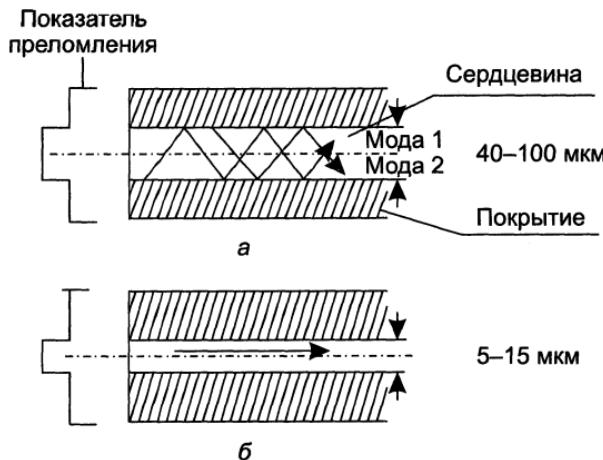


Рис. 2.17. Типы оптического кабеля

В результате многомодовые кабели используются в основном для передачи данных на скоростях не более 1 Гбит/с на небольшие расстояния (до 2000 м), а одномодовые — для передачи данных со сверхвысокими скоростями в несколько десятков гигабитов в секунду на расстояния до нескольких десятков и даже сотен километров (дальняя связь).

В качестве источника света для одномодовых кабелей применяются только лазерные диоды, так как при таком 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 для многомодовых кабелей.

Полоса пропускания и методы кодирования

Изображенные на рис. 2.16 кабели отличаются многими параметрами, например, типом передающей среды (médный провод или оптическое волокно), применяемой электрической и механической изоляцией, типами разъемов и т. п. Поэтому сравнивать кабели различных типов довольно сложно. Для такого сравнения могут быть использованы абстрактные характеристики, выражающие способность кабеля передавать информационные сигналы. Так как указанные характеристики могут относиться не только к кабелю, но и к линии связи любого типа, например, к беспроводному каналу или составному каналу телефонной сети, то мы при их описании будем использовать общий термин «линия связи».

Наиболее важными являются две абстрактные характеристики линии связи:

- затухание (attenuation);**
- полоса пропускания (bandwidth).**

Эти характеристики взаимосвязаны, и для их понимания нужно сначала обратиться к основам спектрального анализа сигналов.

Теория спектрального анализа гласит, что любой *периодический процесс можно представить в виде суммы синусоидальных колебаний различных частот и различных амплитуд* (рис. 2.18).

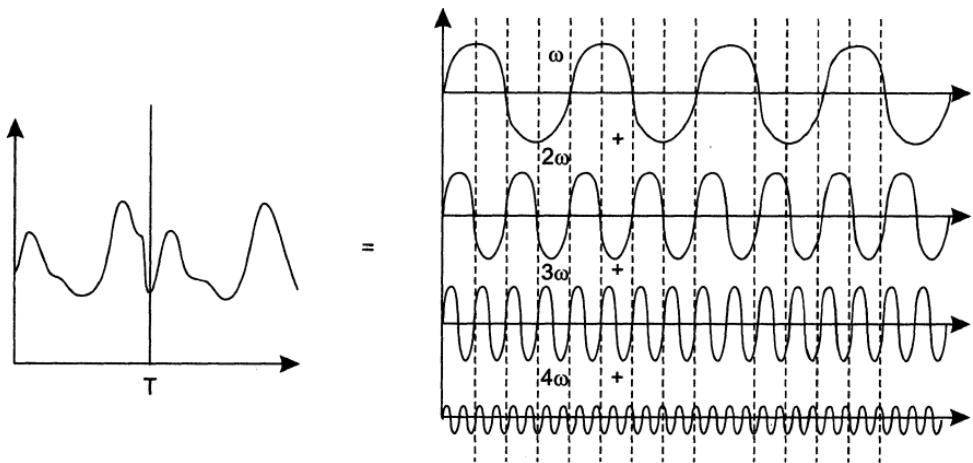


Рис. 2.18. Спектральное представление периодического сигнала

Каждая составляющая синусоида называется также **гармоникой**, а набор всех гармоник называют **спектральным разложением**, или **спектром**, исходного сигнала. Под **шириной спектра сигнала** понимается разность между максимальной и минимальной частотами того набора синусоид, которые в сумме дают исходный сигнал.

Непериодические сигналы можно представить в виде интеграла синусоидальных сигналов с непрерывным спектром частот. 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, распечатывают их на принтере или передают для обработки и хранения в компьютер.

Искажение передающей линией связи синусоиды какой-либо частоты приводит в конечном счете к искажению амплитуды и формы передаваемого сигнала любого вида. Искажения формы проявляются в том случае, если синусоиды 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. 2.19), и сигналы могут плохо распознаваться на приемном конце линии.

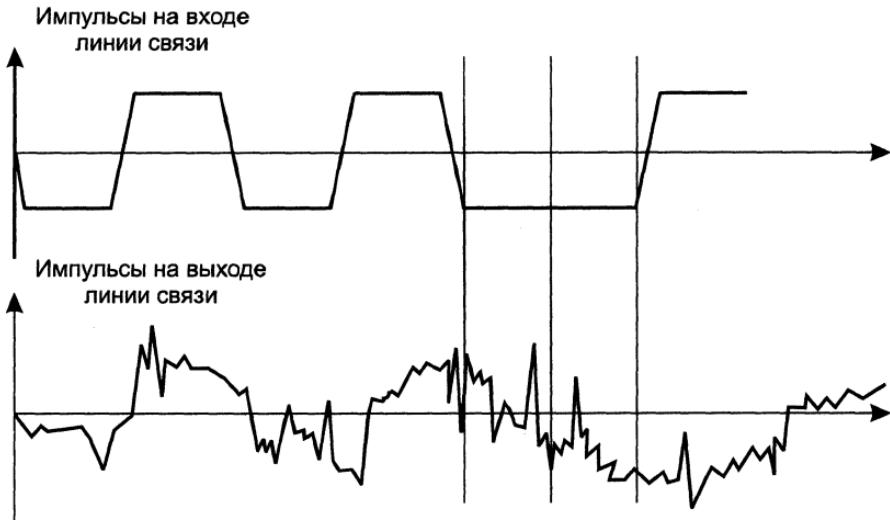


Рис. 2.19. Искажение импульсов в линии связи

Передаваемые сигналы искажаются из-за несовершенства линий связи. Так, идеальная передающая среда на медном проводе, не вносящая никаких помех в передаваемый сигнал, должна, по меньшей мере, иметь нулевые сопротивление, емкость и индуктивность. Однако на практике медные провода, например, всегда представляют собой некоторую распределенную по длине комбинацию активного сопротивления, емкостной и индуктивной нагрузок. В результате синусоиды различных частот передаются этими линиями по-разному.

Помимо искажений сигналов, возникающих из-за не идеальных физических параметров линии связи, существуют и **внешние помехи**, которые вносят свой вклад в искажение формы сигналов на выходе линии. Эти помехи создаются различными электрическими двигателями, электронными устройствами, атмосферными явлениями и т. д. Несмотря на защитные меры, предпринимаемые разработчиками кабелей, и наличие усилительной и коммутирующей аппаратуры, полностью компенсировать влияние внешних помех не удается. Помимо внешних помех в медном кабеле существуют и **внутренние помехи** — так называемые **наводки** одной пары проводников на другую.

Степень искажения сигналов линией связи оценивают с помощью таких характеристик, как затухание и полоса пропускания.

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 (dB) и вычисляется по следующей формуле:

$$A = 10 \lg P_{\text{out}} / P_{\text{in}}.$$

Здесь P_{out} — мощность сигнала на выходе линии, P_{in} — мощность сигнала на входе линии. Так как затухание зависит от длины линии связи, то 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, так как это значение является максимальной длиной кабеля для многих технологий локальных сетей. Для территориальных линий связи погонное затухание измеряют для расстояния в 1 км.

Так как мощность выходного сигнала кабеля без промежуточных усилителей меньше, чем мощность входного сигнала, затухание кабеля всегда является *отрицательной величиной*.

Полоса пропускания линии связи — это диапазон частот, в котором затухание сигнала не превышает некоторый заранее заданный предел.

Это определение подразумевает, что затухание сигнала зависит не только от длины линии связи, но и от частоты. Это в общем случае действительно так, и эта зависимость объясняется различными причинами для различных типов линий связи, например, в линиях на медном проводе распределенная индуктивность и емкость линии зависят от частоты передаваемого сигнала. На рис. 2.20 показаны полосы пропускания линий связи различных типов¹, а также наиболее часто используемые в технике связи частотные диапазоны.

Как видно из рисунка, наиболее широкую полосу пропускания имеют волоконно-оптические кабели, а коаксиальные кабели и кабели на витой паре по этому показателю сопоставимы.

Рисунок 2.21 более детально поясняет природу полосы пропускания оптического волокна. Дело в том, что оптическое волокно характеризуется сложной зависимостью затухания от длины волны, которая имеет три так называемых **окна прозрачности**. Из рисунка видно, что область эффективного использования современных волокон ограничена волнами длиной 850 нм, 1300 нм и 1550 нм (соответственно, 35 ТГц, 23 ТГц и 19,4 ТГц). Окно 1550 нм обеспечивает наименьшие потери и наиболее широкую полосу пропускания, а окно 850 нм — наоборот.

Полоса пропускания является важным показателем линии связи, потому что она непосредственно влияет на ее пропускную способность, которая, в свою очередь, представляет собой главную потребительскую характеристику линии связи.

¹ Обратите внимание, что на рис. 2.20 ось частот имеет логарифмический масштаб.

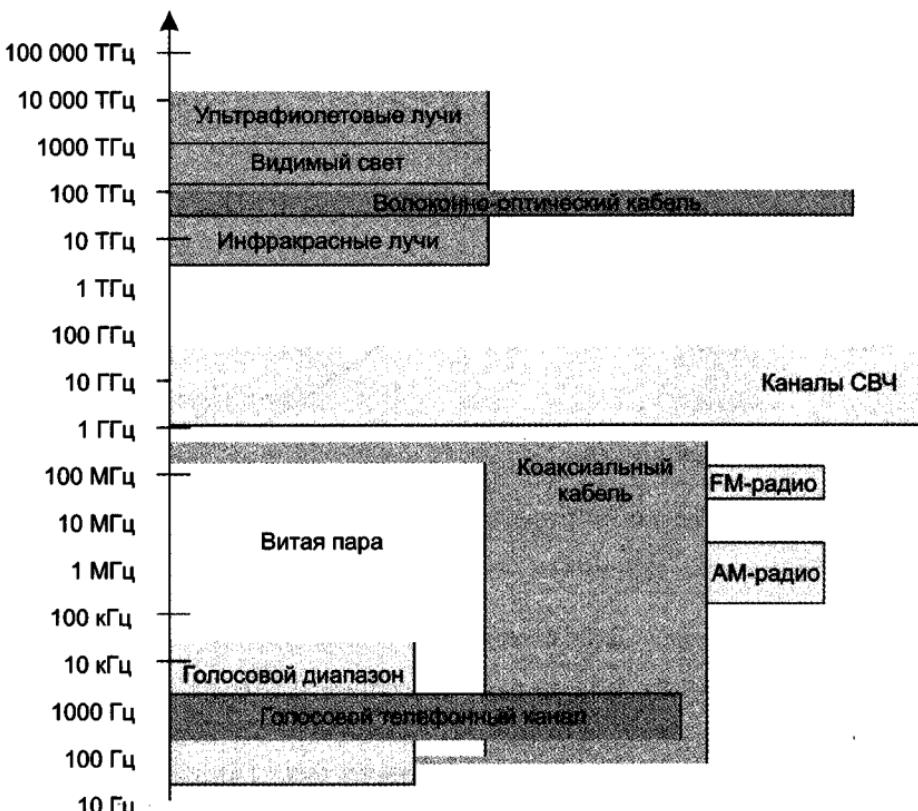


Рис. 2.20. Полосы пропускания линий связи и популярные частотные диапазоны

Пропускная способность линии характеризует максимально возможную скорость передачи данных, которая может быть достигнута на этой линии при фиксированном методе кодирования данных.

Особенностью пропускной способности является то, что, с одной стороны, эта характеристика зависит от *полосы пропускания* линии связи, а с другой — определяется *способом передачи данных*.

ВНИМАНИЕ

Нельзя говорить о пропускной способности линии связи до того, как для нее определен протокол физического уровня.

Например, поскольку для цифровых линий всегда определен протокол физического уровня, задающий битовую скорость передачи данных, то для них всегда известна и пропускная способность — 64 Кбит/с, 2 Мбит/с и т. п. В тех же случаях, когда только предстоит выбрать, какой

из множества существующих протоколов использовать на данной линии, очень важными являются характеристики линии, и в первую очередь ее полоса пропускания.

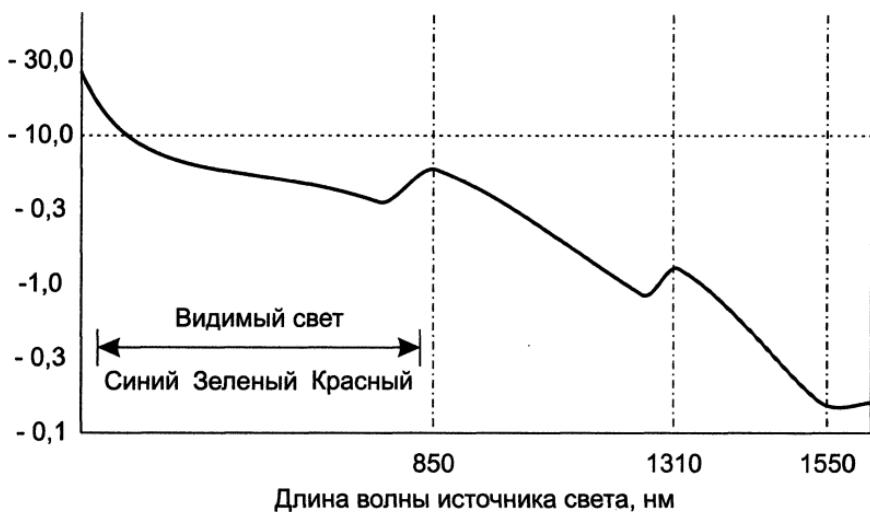


Рис. 2.21. Окна прозрачности оптического волокна

Пропускная способность линии связи зависит как от ее полосы пропускания, так 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. 2.22, а). Если же значимые гармоники выходят за границы полосы пропускания линии связи, то сигнал будет значительно искажаться, а приемник будет ошибаться при распознавании информации (рис. 2.22, б).

В свою очередь, спектр передаваемых сигналов определяется принятым методом представления двоичной информации на линии связи, то есть **методом кодирования**.

При выборе метода кодирования нужно одновременно стремиться к достижению нескольких целей:

- минимизировать ширину спектра сигнала, полученного в результате кодирования;
- обеспечивать синхронизацию между передатчиком и приемником.

Первое требование вытекает из рассмотренной зависимости между полосой пропускания и спектром сигнала. При заданной полосе пропускания более узкий спектр сигнала позволяет добиться более высокой скорости передачи

данных. Это утверждение можно пояснить следующим образом. Спектр сигнала в общем случае зависит как от способа кодирования, так и от тактовой частоты передатчика. Пусть мы разработали два способа кодирования, причем в каждом такте передается один бит информации. Пусть также в первом способе ширина спектра сигнала F равна тактовой частоте смены сигналов f , то есть $F = f$, а второй способ дает зависимость $F = 0,8f$. Тогда при одной и той же полосе пропускания B первый способ позволит передавать данные со скоростью B бит/с, а второй $(1/0,8)B = 1,25 B$ бит/с.

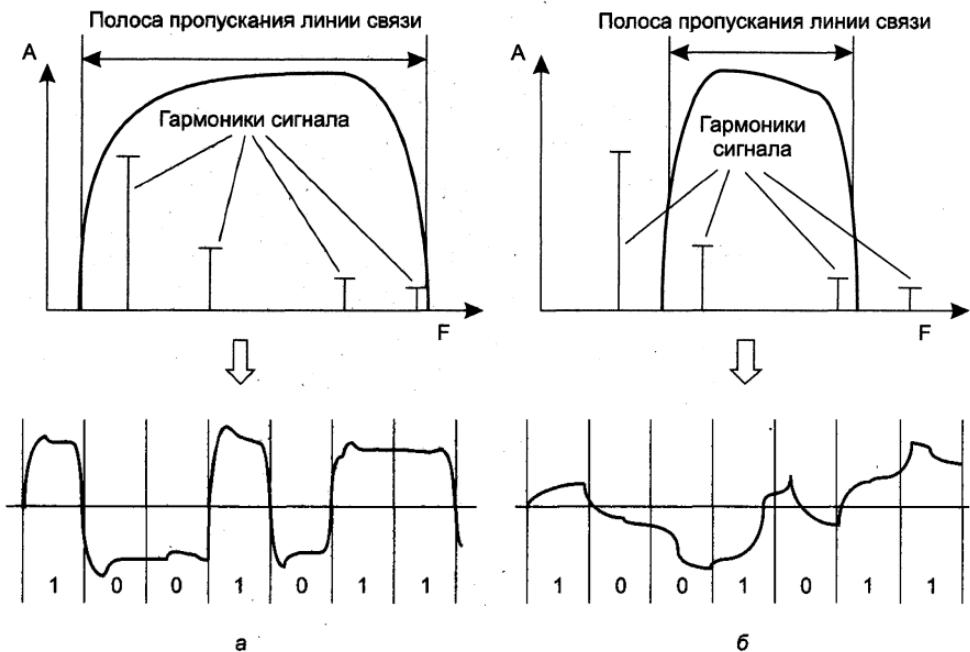


Рис. 2.22. Соответствие между полосой пропускания линии связи и спектром сигнала

Второе требование, то есть обеспечение синхронизации передатчика и приемника, необходимо для того, чтобы приемник точно знал, в какой момент времени считывать новую порцию информации с линии связи. Так как в линиях связи компьютерных сетей не используются дополнительные тактирующие каналы, синхронизация должна обеспечиваться самой передаваемой информацией.

Для решения проблемы синхронизации применяются *самосинхронизирующиеся коды*, сигналы которых несут для приемника указания о том, в какой момент времени нужно осуществлять 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.

Проблема разработчика метода кодирования состоит в том, что описанные требования к коду являются противоречивыми. Действительно, для сужения спектра сигнала (при сохранении той же тактовой частоты) нужно, с одной стороны, уменьшать количество перепадов сигнала в единицу времени. С другой стороны, для повышения синхронизирующих свойств кода наоборот необходимо как можно чаще изменять сигнал, давая приемнику больше шансов на синхронизацию (сравните — постоянный сигнал таких шансов не дает совсем).

Поэтому любой метод кодирования является компромиссом между этими двумя требованиями.

Примером хорошего компромисса может служить *манчестерский код*, с которым мы познакомились при изучении Ethernet на коаксиальном кабеле. Так как сигнал при манчестерском кодировании изменяется по крайней мере один раз в течение такта, то код обладает хорошими самосинхронизирующими свойствами. В то же время основная часть его спектра при тактовой частоте F находится в пределах $2F$, что является неплохим показателем по сравнению с другими кодами.

Классический вариант Ethernet

Мы уже рассмотрели основные характеристики классического варианта технологии Ethernet, в том числе и относящиеся к физическому уровню. Поэтому в данном разделе приводятся только суммарные сведения о стандартах физического уровня, принятых рабочей группой IEEE 802.3 для Ethernet со скоростью 10 Мбит/с.

- **10Base-5** — коаксиальный кабель диаметром 0,5 дюйма, называемый «толстым» коаксиалом. Максимальная длина сегмента — 500 метров (без повторителей). Это первый стандартный вариант физической среды Ethernet, который отличался высокой надежностью, но обладал существенным недостатком — сложностью в прокладке из-за своей ограниченной механической гибкости, а также необходимости располагать приемо-передатчик сетевого адаптера непосредственно на кабеле и соединять его с остальной частью адаптера специальным отрезком витой пары.
- **10Base-2** — коаксиальный кабель диаметром 0,25 дюйма, называемый «тонким» или «дешевым» коаксиалом. Максимальная длина сегмента — 185 метров (без повторителей). Этот стандарт вытеснил «толстый» коаксиал благодаря своей дешевизне и простоте прокладки кабеля, так как он позволял подводить кабель непосредственно к Т-образным разъемам сетевого адаптера компьютера и легко добавлять новые компьютеры к существующему сегменту сети (рис. 2.23).
- **10Base-T** — кабель на основе неэкранированной витой пары (UTP). Образует звездообразную топологию на основе концентратора (хаба) или коммутатора. Расстояние между портами 10Base-T (например, между

портами коммутаторов или портом сетевого адаптера и коммутатора) — не более 100 м.

- **10Base-T** — волоконно-оптический кабель. Топология аналогична топологии стандарта 10Base-T. Имеется несколько вариантов этой спецификации: FOIRL (расстояние сегмента до 1000 м), 10Base-FL (расстояние до 2000 м), 10Base-FB (расстояние до 2000 м). Увеличение максимальной длины сегмента по сравнению с другими вариантами 10 Мбит/с Ethernet объясняется гораздо более широкой полосой пропускания волоконно-оптического кабеля.

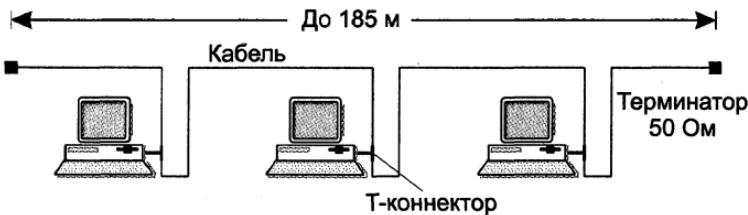


Рис. 2.23. Сеть стандарта 10Base-2

Число 10 в перечисленных названиях обозначает номинальную битовую скорость передачи данных этих стандартов — 10 Мбит/с, а слово «Base» — метод передачи на одной базовой частоте 10 МГц; этим они отличаются от стандартов, использующих несколько несущих частот (они называются широкополосными и имеют в своем составе слово «Broadband»). Последний символ в названии стандарта физического уровня обозначает тип кабеля.

Сегодня из всех стандартов 10 Мбит/с Ethernet применяется только 10Base-T — в основном в тех случаях, когда скорость передачи данных не является критичной, например, при соединении порта SNMP коммутатора или маршрутизатора с системой управления.

Стандарт 10Base-F вышел из употребления по той причине, что оптическое волокно может поддерживать гораздо более высокие скорости, чем 10 Мбит/с, и в тех случаях, когда имеется возможность или необходимость использовать волоконно-оптический кабель, рациональнее применять высокоскоростные версии Ethernet.

Fast Ethernet

Осеню 1995 года после 3 лет обсуждений комитет IEEE 802.3 в окончательном виде одобрил стандарт **Fast Ethernet**. Этот стандарт описывает три варианта физического уровня:

- 100Base-FX для волоконно-оптического кабеля с двумя волокнами;
- 100Base-T4 для четырехпарного кабеля на витой паре;
- 100Base-TX для двухпарного кабеля на витой паре.

Волоконно-оптический кабель

Вариант **100Base-FX** определяет в качестве среды передачи данных много-модовый оптический кабель и волну 850 нм, что обеспечивает связь между портами двух коммутаторов или маршрутизаторов на расстоянии до 2000 м. Одномодовый оптический кабель в стандарте 100Base-FX не описывается, тем не менее на рынке можно найти оборудование Fast Ethernet, работающее и на таком типе кабеля; при этом максимальная длина одного сегмента кабеля может доходить до нескольких десятков километров.

Кабель на витой паре категории 5

Наибольшие сложности для десятикратного повышения битовой скорости возникли у разработчиков нового стандарта при создании версии Ethernet на витой паре. Как известно, успех стандарта 10Base-T был обеспечен тем, что он опирался на уже имеющуюся во многих зданиях телефонную проводку, состоящую из 4 витых пар. Согласно стандарту 10Base-T, для передачи данных используются только две витые пары, остающиеся в кабеле свободными после подключения телефона, так что новая специальная проводка для компьютерной сети не требуется, если в здании уже есть телефонная.

Проблема заключалась в том, что телефонная проводка с начала 90-х годов выполнялась витой парой категории 3, имеющей полосу пропускания до 16 МГц, что вполне подходило как для качественной передачи голоса, так и для передачи данных со скоростью 10 Мбит/с, так как при манчестерском кодировании основная часть спектра сигнала не выходит за пределы этой полосы.

Однако для скорости 100 Мбит/с такой полосы пропусканияказалось недостаточно, даже несмотря на усилия разработчиков по усовершенствованию методов кодирования.

В результате было принято решение о стандартизации двух вариантов Fast Ethernet на витой паре:

- 100Base-T4** – вариант для существующей витой пары категории 3, но использующий для передачи данных сразу 4 пары;
- 100Base-TX** – специально разработанный и стандартизованный для Fast Ethernet вариант, работающий на двух витых парах более качественного кабеля категории 5.

Стандарт 100Base-T4 оказался промежуточным компромиссным вариантом, не нашедшим широкого применения, так как промышленность быстро освоила выпуск кабелей категории 5, которые оказались ненамного дороже кабелей категории 3.

Полоса пропускания кабелей категории 5 покрывает диапазон частот до 100 МГц, что позволило повысить битовую скорость до 100 Мбит/с без дополнительных пар. Помимо более широкой полосы пропускания, кабель категории 5 имеет еще несколько преимуществ по сравнению с кабелем

категории 3 и в первую очередь — более высокую степень защиты пар от взаимных помех, которые возникают в результате наводок электромагнитного поля в одной паре при передаче данных по другой паре.

Свой вклад в повышение скорости Ethernet внесли и разработчики методов кодирования.

Код 4B/5B

В версиях 100Base-TX и 100Base-FX для представления двоичных данных используется код 4B/5B, обладающий более узким спектром, чем манчестерский, и, следовательно, позволяющий передавать данные с более высокой скоростью на кабеле с фиксированной полосой пропускания.

Код 4B/5B основан на потенциальном коде NRZI (рис. 2.24).

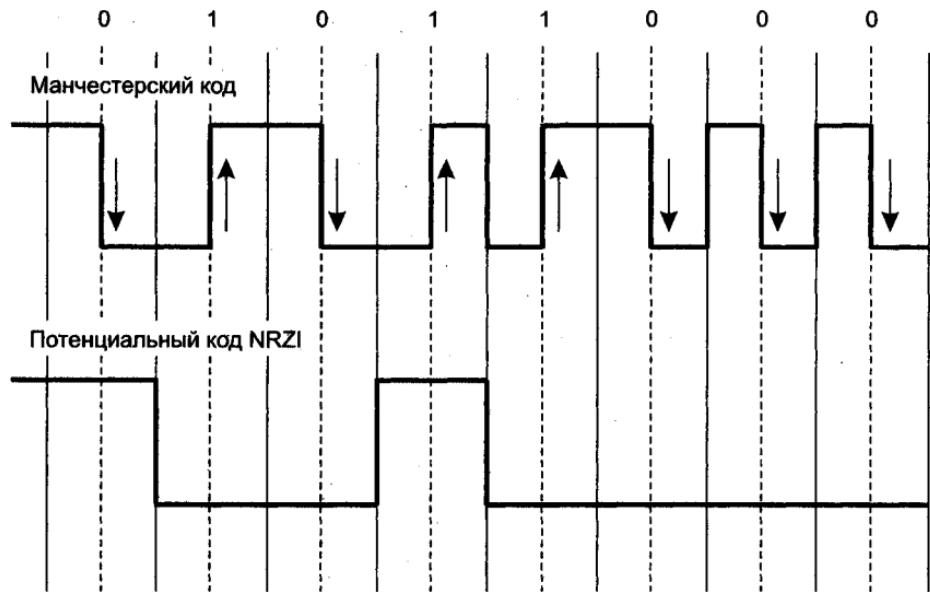


Рис. 2.24. Сравнение кода NRZI и манчестерского кода

В коде NRZI (Non Return to Zero with ones Inverted — без возвращения к нулю и с инверсией при единице) для кодирования двоичных единиц и нулей используются два потенциала электрического сигнала (или же периоды света и темноты в случае оптических сигналов). Однако эти потенциалы не закреплены жестко за значением двоичной информации, вместо этого код NRZI сохраняет потенциал предыдущего такта, если на текущем такте передается нуль, и инвертирует его, если передается единица.

Код NRZI хорош тем, что в среднем он требует меньше изменений сигнала при передаче произвольной двоичной информации, чем манчестерский код, за счет чего спектр его сигналов уже. Однако код NRZI обладает плохой

самосинхронизацией, так как при передаче длинных последовательностей нулей сигнал вообще не меняется (например, при передаче последних 3 нулей на рис. 2.24) и, значит, у приемника исчезает возможность синхронизации с передатчиком на значительное время, что может приводить к ошибкам распознавания данных.

Код 4B/5B решает проблему плохой самосинхронизации кода NRZI за счет введения избыточности, то есть код 4B/5B принадлежит к семейству так называемых избыточных кодов.

Избыточные коды основаны на разбиении исходной последовательности битов на порции, которые часто называют *символами*, после чего каждый исходный символ заменяется новым с большим количеством битов, чем исходный.

Например, в избыточном коде **4B/5B**, впервые использованном в технологии FDDI, а затем позаимствованном 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. Поэтому в результирующем коде можно отобрать 16 таких комбинаций, которые не содержат большого количества нулей, а остальные считать **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.

После разбиения получившийся код 4B/5B передается по линии путем преобразования методом потенциального кодирования, такого как NRZI, чувствительного только к длинным последовательностям нулей. Символы кода 4B/5B длиной 5 бит гарантируют, что при любом их сочетании на линии не встретятся более трех нулей подряд.

ПРИМЕЧАНИЕ

Буква В (от английского *binary* — двоичный) в названии кода 4B/5B означает, что элементарный сигнал имеет 2 состояния.

Использование таблицы перекодировки является очень простой операцией, поэтому этот подход не усложняет сетевые адаптеры и интерфейсные блоки 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 передатчик должен работать с тактовой частотой 125 МГц. При этом спектр сигнала на линии расширяется по сравнению с передачей не избыточного кода. Тем

не менее спектр избыточного потенциального кода оказывается уже спектра манчестерского кода, что оправдывает дополнительный этап логического кодирования, а также работу приемника и передатчика на повышенной тактовой частоте.

Режим автопереговоров

Стандарт Fast Ethernet включает в себя **режим автопереговоров**, который позволяет двум физически соединенным устройствам, которые поддерживают несколько стандартов физического уровня, различающихся битовой скоростью и количеством витых пар, согласовать наиболее выгодный режим работы. Обычно процедура автопереговоров происходит при подсоединении сетевого адаптера, который может работать на скоростях 10 и 100 Мбит/с, к концентратору или коммутатору.

Всего в настоящее время определено 5 различных режимов работы, которые могут поддерживать устройства 100Base-TX/T4 на витых парах:

- 10Base-T;
- дуплексный режим 10Base-T;
- 100Base-TX;
- 100Base-T4;
- дуплексный режим 100Base-TX.

Режим 10Base-T имеет самый низкий приоритет в переговорном процессе, а дуплексный режим 100Base-T4 – самый высокий.

Переговорный процесс происходит при включении питания устройства, а также может быть инициирован в любой момент модулем управления устройства. Устройство, начавшее процесс автопереговоров, посыпает своему партнеру пачку специальных импульсов **FLP** (Fast Link Pulse), в которой содержится 8-битное слово, кодирующее предлагаемый режим взаимодействия, начиная с самого приоритетного, поддерживаемого данным узлом.

Если узел-партнер способен к автопереговорам и также может поддерживать предложенный режим, он отвечает пачкой импульсов FLP, в которой подтверждает данный режим, и на этом переговоры заканчиваются. Если же узел-партнер не может поддерживать запрошенный режим, то он указывает в своем ответе имеющийся в его распоряжении следующий по степени приоритетности режим, и этот режим выбирается в качестве рабочего.

Порты, поддерживающие режим автопереговоров, часто имеют маркировку 10/100 Ethernet.

Gigabit Ethernet

Достаточно быстро 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 сетевые магистрали FDDI и Fast Ethernet, также работающие на скорости 100 Мбит/с. Ощущалась потребность в следующем уровне иерархии скоростей. В 1995 году более высокий уровень скорости могли предоставить только коммутаторы ATM, которые из-за высокой стоимости, а также больших отличий от классических технологий применялись в локальных сетях достаточно редко.

Поэтому логичным выглядел следующий шаг, сделанный IEEE. Летом 1996 года было объявлено о начале разработки протокола, в максимальной степени подобного Ethernet, но с битовой скоростью 1000 Мбит/с. Проблемная группа IEEE 802.3ab успешно справилась со своей задачей, и версия **Gigabit Ethernet** для витой пары категории 5 была принята.

ПРИМЕЧАНИЕ

Достаточно много проблем у разработчиков Gigabit Ethernet вызвало стремление сохранить возможность работы на разделяемой среде. Эти проблемы были решены, однако, как показала практика, усилия оказались напрасными, так как Gigabit Ethernet на разделяемой среде практически никогда не применялся.

В стандарте Gigabit Ethernet определены следующие варианты физического уровня.

- **1000Base-LX** допускает работу как с многомодовым (максимальное расстояние до 500 м), так и с одномодовым волоконно-оптическим кабелем (максимальное расстояние зависит от мощности передатчика и качества кабеля и может доходить до нескольких десятков километров).
- **1000Base-SX** позволяет использовать только многомодовый волоконно-оптический кабель, при этом его максимальная длина составляет около 500 м.
- **1000Base-T** допускает работу на витой паре UTP категорий 5 и 6. Для работы на скорости 1000 Мбит/с на витой паре категории 5 в стандарте Gigabit Ethernet используются все 4 пары кабеля. Максимальная длина кабеля такая же, как и в других стандартах Ethernet на витой паре, то есть 100 м. Данные передаются параллельно по всем парам со скоростью 250 Мбит/с по каждой из них. При этом поддерживается передача в дуплексном режиме, то есть одновременно в обоих направлениях. Кабель категории 6 обладает лучшими характеристиками, чем кабель категории 5.
- **1000Base-CX** формально вернул коаксиальный кабель в состав поддерживаемых сред передачи данных, но на практике этот вариант с максимальной длиной сегмента 25 м используется редко.

Для организации дуплексного режима в спецификации 1000Base-T разработчики стандарта применили технику выделения принимаемого сигнала из суммарного. Два передатчика работают навстречу друг другу по каждой из 4 пар в одном и том же диапазоне частот (рис. 2.25).

Для отделения принимаемого сигнала от собственного приемник вычитает из результирующего сигнала известный ему свой сигнал. Естественно, что это не простая операция и для ее выполнения используются специальные процессоры цифрового сигнала (Digital Signal Processor, DSP). Для поддержания скорости 250 Мбит/с в кабеле категории 5 был разработан новый код **PAM5**, который вместо двух значений потенциала электрического сигнала использует пять.

Как и Fast Ethernet, Ggabit Ethernet поддерживает процедуру автопереговоров.

Для удобства перехода с одной среды передачи данных на другую порты Gigabit Ethernet имеют сменные приемопередатчики, так называемые модули **GBIC** (Gigabit Interface Converter — конвертор гигабитного интерфейса). Используя такие модули, один и тот же порт Gigabit Ethernet может работать с любой из стандартных сред, для этого нужно приобрести и установить соответствующий кабель модуль GBIC. Существует компактная версия GBIC, называемая **SFP** (Small Form-factor Pluggable), которая функционально эквивалентна GBIC.

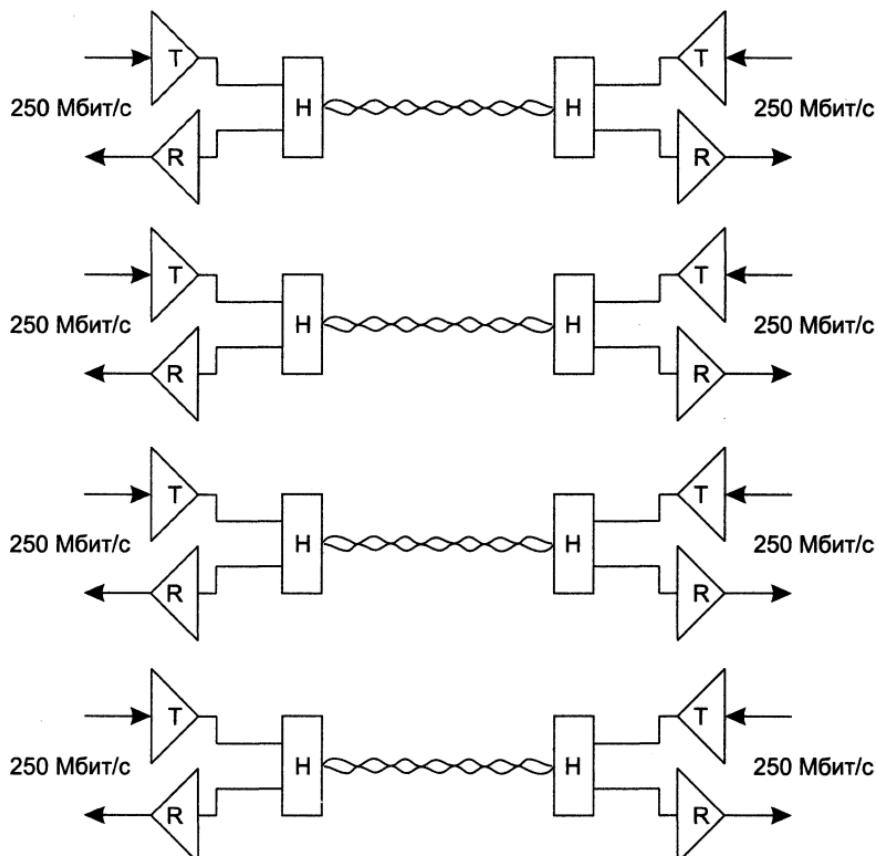


Рис. 2.25. Двунаправленная передача по четырем парам кабеля категории 5

10G Ethernet

На момент написания книги стандарт **10G Ethernet** представлял собой самый скоростной вариант технологии Ethernet. Это первый стандарт Ethernet, который не работает на разделяемой среде даже теоретически. Кроме того, это первый стандарт Ethernet, который включает спецификации физического уровня, совместимые со стандартами глобальных сетей, а именно, сетей SDH (они рассматриваются в главе 4).

Стандарт 10G Ethernet включает большое количество различных спецификаций физического уровня. Первая группа таких спецификаций, рассчитанная на использование оптического волокна, была принята в 2002 году, когда завершилось создание ядра этого стандарта. Однако и после этого работы продолжались, и в 2006 году был принята спецификация, описывающая функционирование 10G Ethernet на витой паре.

Существует три группы физических интерфейсов стандарта 10G Ethernet.

- **10GBase-T** принят в 2006 году, он дает возможность использовать знакомые администраторам локальных сетей кабели на витой паре. Правда, обязательным требованием является применение кабелей категории 6 или 6a: в первом случае максимальная длина кабеля не должна превышать 55 м, во втором — 100 м, что является традиционным для локальных сетей.
- **10Gbase-R** работает на оптическом кабеле, включает спецификации 10GBase-RS, 10GBase-RL, 10GBase-RE.
- **10GBase-W** работает на оптическом кабеле, включает спецификации 10GBase-WS, 10GBase-WL, 10GBase-WE.

Первые две группы относятся к так называемому варианту 10G Ethernet для локальных сетей, в то время как последняя — к варианту для глобальных сетей.

Различие заключается в том, что в версиях для локальных сетей используются стандартные кадры Ethernet и обеспечивается битовая скорость передачи данных 10 Гбит/с. Версии 10G Ethernet для глобальных сетей разработаны в расчете на первичные сети SDH и поддерживают скорость передачи и формат данных, совместимые с интерфейсом сетей SDH. Из-за этого эффективная скорость передачи данных спецификаций для глобальных сетей несколько ниже 10 Гбит/с (она равна 9,58464 Гбит/с), так как часть пропускной способности тратится на заголовки кадров SDH. Из-за того, что скорость передачи информации у этой группы интерфейсов ниже, чем 10 Гбит/с, они могут взаимодействовать только между собой, то есть соединение, например, интерфейсов 10GBase-R и 10Base-W невозможно.

В каждой из групп 10GBase-W и 10GBase-R, которые работают на оптическом кабеле, может быть три варианта спецификаций: S, L и E (в зависимости от используемого для передачи информации диапазона волн: 850, 1310 или 1550 нм соответственно). Таким образом, существуют интерфейсы

10GBase-WS, 10GBase-WL, 10GBase-WE, а также 10GBase-RS, 10GBase-RL и 10GBase-RE. Каждый из них передает информацию с помощью одной волны соответствующего диапазона.

Спецификации S рассчитаны на многомодовый оптический кабель длиной до 300 м в зависимости от качества кабеля. Спецификации L рассчитаны на одномодовый кабель и в зависимости от его качества допускают расстояния до 25 км. Физические интерфейсы, работающие в окне прозрачности E, обеспечивают передачу данных на расстояния до 40 км. Это позволяет строить не только локальные, но и глобальные сети.

Стандарт 10G Ethernet является развивающейся технологией, так что можно ожидать появление его новых спецификаций. В настоящее время ведется работа над двумя новыми стандартами Ethernet: 40G и 100G, которые должны найти свое применение на магистралях крупных сетей.

Виртуальные локальные сети

Пользовательские фильтры

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 эти узлы подключены к одной и той же локальной сети. В то же время в сети могут возникать ситуации, когда такая тотальная доступность узлов нежелательна. Примером может служить сервер финансового отдела, доступ к которому желательно разрешить только с компьютеров нескольких конкретных сотрудников этого отдела. Конечно, доступ можно ограничить на уровне операционной системы или системы управления базой данных самого сервера, но для обеспечения высокой надежности желательно иметь несколько эшелонов защиты и ограничить доступ еще и на уровне передачи сетевого трафика.

Одним из возможных вариантов решения этой задачи на уровне сетевых устройств является применение в коммутаторах сети пользовательских фильтров.

Пользовательский фильтр, который также часто называют **списком доступа** (access lists), – это набор условий, которые ограничивают обычную логику передачи кадров коммутаторами.

Рассмотрим его применение на примере сети, показанной на рис. 2.26.

Пусть мы хотим разрешить доступ к серверу S1 только компьютерам C1 и C3, кадры от всех остальных компьютеров до этого сервераходить не должны. 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 для того, чтобы в дальнейшем была возможность добавить в этот список другие записи, сохранив исходную последовательность строк. Первое условие разрешает (permit) передачу кадра, если его адрес источника равен MAC-C1, а адрес назначения — MAC-S1; второе условие делает то же, но для кадра с адресом источника MAC-C3, третье условие запрещает (deny) передачу кадров с любыми (any) адресами.

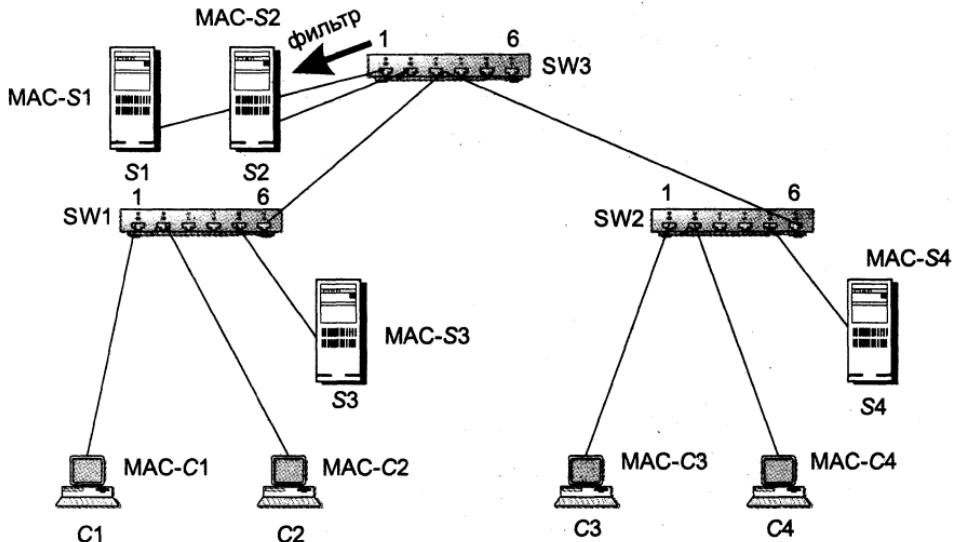


Рис. 2.26. Контроль доступа к серверу с помощью пользовательского фильтра

Для того чтобы список доступа начал работать, его нужно применить к какому-либо порту коммутатора в определенном направлении: либо для входящего трафика, либо для исходящего. В нашем примере нам нужно применить список доступа к порту 1 коммутатора SW3, к которому подключен сервер S1 для исходящего трафика. Коммутатор SW3, перед тем как передать кадр на порт 1, будет просматривать условия списка доступа по очереди. Если какое-то условие из списка выполняется, то коммутатор выполняет действие этого условия над обрабатываемым кадром, и на этом применение списка доступа для данного кадра заканчивается.

Поэтому, когда от компьютера C1 приходит кадр, адресованный серверу S1, выполняется первое условие списка, которое разрешает передачу кадра, так что коммутатор выполняет стандартное действие по продвижению кадра и тот доходит до сервера S2. С кадром от компьютера C3 совпадение

происходит при проверке второго условия, и он также передается. Однако когда приходят кадры от других компьютеров, например, компьютера С2, то ни первое, ни второе условие не выполняется, а выполняется третье условие, поэтому кадр не передается, а отбрасывается.

Списки доступа коммутаторов не работают для широковещательного адреса Ethernet, такие кадры всегда передаются на все порты коммутатора. Списки доступа коммутаторов достаточно примитивны, так как они могут оперировать только с информацией канального уровня, то есть MAC-адресами. Списки доступа маршрутизаторов гораздо более гибкие и мощные, поэтому на практике они применяются намного чаще.

Логическое разделение сети на виртуальные локальные сети

Как мы видели на примере из предыдущего раздела, 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 группы.
- ❑ *Невозможно блокировать широковещательный трафик*. Из-за этого свойства сети, созданные на основе коммутаторов, иногда называют *плоскими* — так как в них отсутствуют барьеры на пути широковещательного трафика. 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.

Виртуальной локальной сетью (Virtual LAN, VLAN) называется группа узлов, 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. Говорят, что виртуальная сеть образует *домен широковещательного трафика*, поскольку широковещательный трафик не выходит за пределы соответствующей группы узлов.

Основное назначение технологии VLAN состоит в создании логически изолированных сетей внутри одной и той же физически связанной локальной сети. Эти логически изолированные сети могут вообще не общаться между собой — такой вариант в принципе возможен, если, например, предприятию по каким-то соображениям требуется создать полностью защищенную от внешнего мира сеть.

На практике же виртуальные локальные сети чаще всего между собой общаются, но только через маршрутизатор, который рассматривает их как физически разные сети. Как уже было отмечено, маршрутизатор является более сложным и гибким устройством и может более тонко ограничивать взаимодействие между сетями. Кроме того, маршрутизатор не передает между своим портами широковещательный трафик канального уровня, что автоматически решает проблему затопления всей сети каким-то одним неправильно работающим узлом.

Достоинством технологий виртуальных сетей является то, что она позволяет создавать полностью изолированные сегменты сети путем логического конфигурирования коммутаторов и без изменения физической структуры сети.

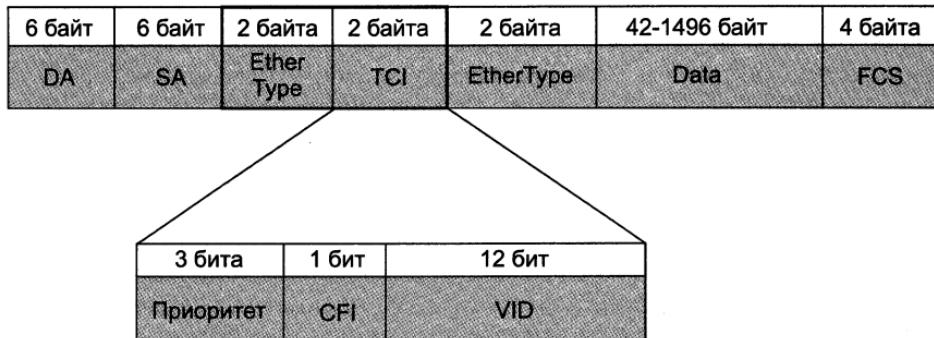
Технология виртуальных сетей долгое время не стандартизовалась, хотя и была реализована в очень широком спектре моделей коммутаторов разных производителей. Положение изменилось после принятия в 1998 году стандарта **IEEE 802.1Q**, который определяет базовые правила построения виртуальных локальных сетей.

Этот стандарт вводит в кадре Ethernet дополнительный заголовок, который называется VLAN-тегом.

VLAN-тег состоит из поля **TCI** (Tag Control Information — управляющая информация тега) размером 2 байта и предшествующего ему поля **EtherType** (рис. 2.27).

VLAN-тег 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.

Для того 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 VLAN-тега следует другое поле, **EtherType**, определяющее тип протокола, данные которого переносятся полем данных кадра.

VLAN тег**Рис. 2.27.** Структура помеченного кадра Ethernet

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 3-битное поле *приоритета* кадра. Однобитное поле *CFI* было введено с целью поддержания специального формата кадра Token Ring, для сетей Ethernet оно должно содержать значение 0.

Структура помеченного кадра Ethernet показана на рис. 2.27. Из-за добавления тега максимальная длина поля данных уменьшилась на 4 байта.

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. Эта процедура называется также **группировкой портов**.

Для упрощения конфигурирования сети стандарт 802.1Q вводит понятия линии доступа и транка.

- **Линия доступа** связывает порт коммутатора (называемый в этом случае **портом доступа**) с компьютером, принадлежащим некоторой виртуальной локальной сети.
- **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.

Для того 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, присвоить к этой сети те порты, к которым присоединены включаемые в нее компьютеры. Порт доступа может быть присвоен только к одной виртуальной локальной сети.

Порты доступа получают от конечных узлов сети непомеченные кадры и помечают их тегом виртуальной локальной сети, 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. 2.28 показано, как решается задача избирательного доступа к серверам на основе техники VLAN.

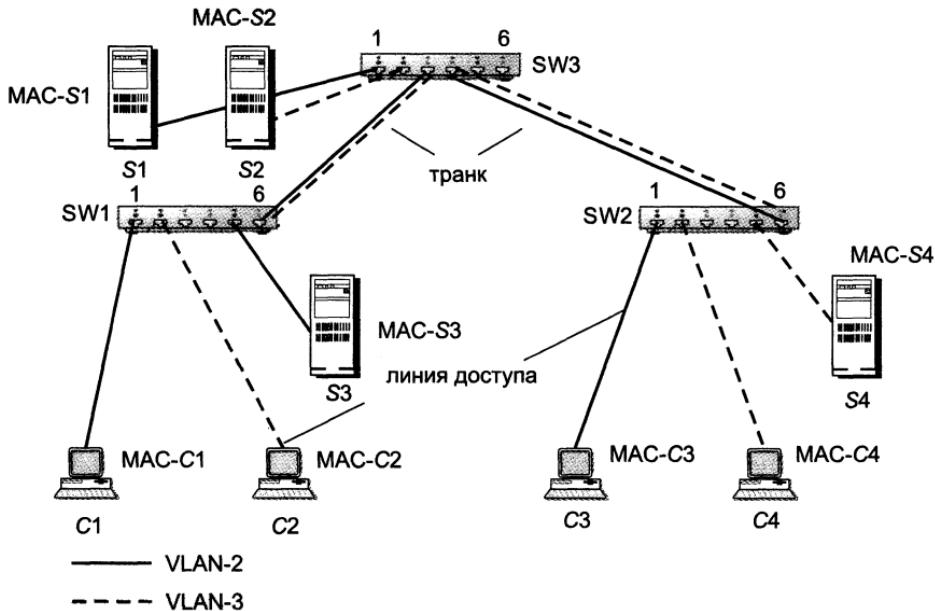


Рис. 2.28. Разбиение сети на две виртуальные локальные сети

Будем считать, что поставлена задача обеспечить доступ компьютеров *C1* и *C3* к серверам *S1* и *S3*, в то время как компьютеры *C2* и *C4* должны иметь доступ только к серверам *S2* и *S4*.

Чтобы решить эту задачу можно организовать в сети две виртуальные локальные сети, VLAN2 и VLAN3 (напомним, что сеть VLAN1 уже существует по умолчанию — это наша исходная сеть), присвоив один набор компьютеров и серверов к VLAN2, а другой — к VLAN3.

Для приписывания конечных узлов к определенной сети VLAN соответствующие порты объявляются портами доступа этой виртуальной локальной 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, они 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 VLAN2 и VLAN3 (и VLAN1, если в сети есть узлы, явно не приписанные ни к одной виртуальной локальной сети).

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 кадра виртуальной локальной сети, приписанной 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.

Как мы видим из примера, техника VLAN оказывается весьма эффективной для разграничения доступа к серверам. Конфигурирование виртуальной локальной сети не требует знания MAC-адресов узлов, кроме того, любое изменение в сети, например подключение компьютера к другому коммутатору, требует конфигурирования лишь порта данного коммутатора, а все остальные коммутаторы сети продолжают работать без внесения изменений в их конфигурации.

По умолчанию протокол STP/RSTP образует в сети одно покрывающее дерево для всех виртуальных локальных сетей. Чтобы в каждой такой сети можно было использовать собственное покрывающее дерево, существует специальная версия протокола, называемая **множественным протоколом STP** (**Multiple STP, MSTP**).

Беспроводные локальные сети

Беспроводные локальные сети (Wireless LAN, WLAN) в некоторых случаях являются предпочтительным по сравнению с проводной сетью решением, а иногда просто единственным возможным. В WLAN сигнал распространяется

с помощью электромагнитных волн высокой частоты. Примеры популярных областей применения WLAN:

- организация так называемого «кочевого» доступа в аэропортах и железнодорожных вокзалах;
- создание временных локальных сетей, например, при проведении конференций;
- реализация доступа в Интернет в жилых домах и квартирах, где кабельная проводка создает неудобства;
- обеспечение мобильного доступа в пределах нескольких помещений или зданий, что очень актуально, например, для больниц.

Однако за преимущества беспроводных сетей приходится расплачиваться множеством проблем, которые несет с собой неустойчивая и непредсказуемая беспроводная среда. Помехи от разнообразных бытовых приборов и других телекоммуникационных систем, атмосферные помехи и отражения сигнала создают большие трудности для надежного приема информации. Локальные сети — это, прежде всего, сети зданий, а распространение радиосигнала внутри здания подвержено влиянию многих факторов. На рис. 2.29 показан пример распределения интенсивности радиосигнала. Подчеркнем, что это статическое изображение, в действительности же картина является динамической, и при перемещении объектов в комнате распределение сигнала может существенно измениться.

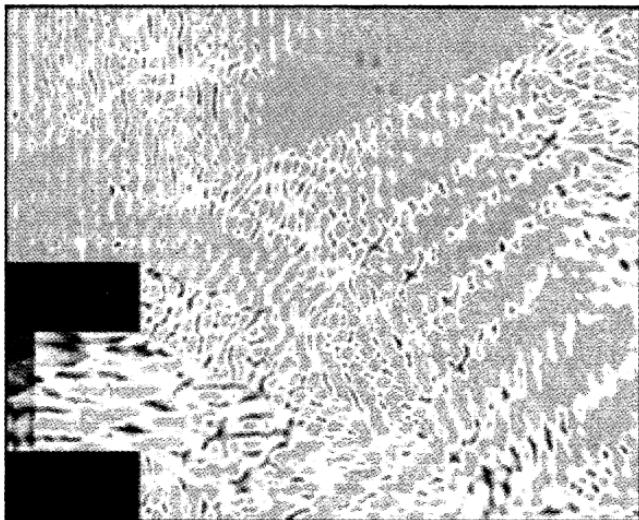


Рис. 2.29. Распределение интенсивности радиосигнала в комнате

В связи с этим в WLAN применяются сложные методы кодирования, которые помогают снизить влияние помех на полезный сигнал, кроме того, в беспроводных сетях широко используются методы прямой коррекции ошибок

(Forward Error Control, FEC) и протоколы с повторной передачей потерянных кадров.

Неравномерное распределение интенсивности сигнала приводит не только к битовым ошибкам передаваемой информации, но и к *неопределенности зоны покрытия* беспроводной локальной сети.

Часто используемое изображение такой области в форме шестиугольника или круга является не чем иным, как абстракцией. В действительности сигнал может быть настолько ослаблен, что устройства, находящиеся в предполагаемых пределах зоны покрытия, вообще не могут принимать и передавать информацию.

Стандарты IEEE 802.11

В 1997 году комитетом **IEEE 802.11** был принят стандарт, который определял функции беспроводной локальной сети, а также работу *трех вариантов физического уровня*, обеспечивающих передачу данных со скоростями 1 и 2 Мбит/с. С тех пор стандарты 802.11 стали основными для беспроводных локальных сетей, они постепенно дополняются новыми спецификациями физического уровня, которые уже расширили диапазон скоростей передачи данных в WLAN до 52 Мбит/с (табл. 2.1). Все варианты стандарта 802.11 поддерживают один и тот же алгоритм доступа к разделяемой среде, которой является радиоэфир.

Таблица 2.1. Физические уровни стандарта 802.11

802.11		802.11a	802.11b	802.11g
2,4 ГГц 1 Мбит/с 2 Мбит/с	Инфракрасные волны 1 Мбит/с 2 Мбит/с	5 ГГц До 54 Мбит/с	2,4 ГГц До 11 Мбит/с	2,4 ГГц До 54 Мбит/с

На момент написания этих строк (2008 год) разрабатывался новый стандарт физического уровня 802.11n, которыйставил своей целью повышение битовой скорости обмена до 600 Мбит/с. Оборудование, соответствующее предварительной версии этого стандарта, обеспечивало скорости обмена до 300 Мбит/с.

Максимальный диаметр сети 802.11 зависит от многих параметров, в том числе и от диапазона частот. Обычно он находится в пределах от 100 до 300 м.

Стандарт 802.11 поддерживает два типа топологий локальных сетей: с базовым и с расширенным наборами услуг.

Сеть с базовым набором услуг (Basic Service Set, BSS) образуется отдельными станциями, которые взаимодействуют друг с другом непосредственно (рис. 2.30).

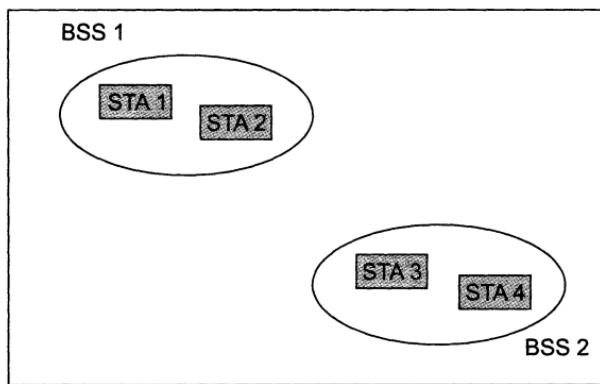


Рис. 2.30. Сети с базовым набором услуг

Сети BSS не являются традиционными сотами в отношении зон покрытия, они могут находиться на значительном расстоянии друг от друга, а могут частично или полностью перекрываться — стандарт 802.11 оставляет здесь свободу для проектировщика сети.

Сеть с расширенным набором услуг (Extended Service Set, ESS) состоит из нескольких сетей BSS, объединенных распределенной средой (рис. 2.31).

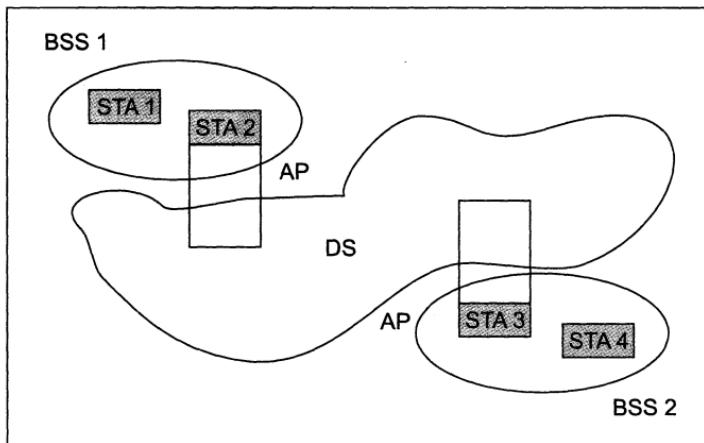


Рис. 2.31. Сети 802.11 с расширенным набором услуг

В сетях ESS некоторые станции сети являются базовыми (в отличие от остальных, называемых рабочими), или, в терминологии 802.11, **точками доступа** (Access Point, AP). Станция, которая выполняет функции AP, является членом какой-нибудь сети BSS. Все базовые станции сети связаны между собой с помощью распределенной системы (Distribution System, DS), в качестве которой может использоваться та же среда, что и для взаимодействия между станциями (то есть радио- или инфракрасные волны), или же отличная от нее, например проводная.

Точки доступа вместе с распределенной системой поддерживают **службу распределенной системы** (Distribution System Service, DSS). Задачей DSS является передача пакетов между станциями, которые по каким-то причинам не могут или не хотят взаимодействовать между собой непосредственно. Наиболее очевидной причиной использования DSS является принадлежность станций разным сетям BSS. В этом случае они передают кадр своей точке доступа, которая через DS передает его точке доступа, обслуживающей сеть BSS станции назначения.

Сеть ESS придает станциям мобильность — они могут переходить из одной сети BSS в другую. Кроме того, ESS может также взаимодействовать с проводной локальной сетью.

В беспроводных сетях стандарта 802.11 поддерживаются два режима доступа к разделяемой среде:

- распределенный режим;
- централизованный режим.

Распределенный режим (Distributed Coordination Function, DCF) вместо алгоритма распознавания коллизий CSMA/CD, принятого в проводных сетях Ethernet, использует алгоритм предотвращения коллизий, называемый **CSMA/CA** (две последние буквы аббревиатуры взяты от Collision Avoidance — дословно «избежание коллизий»). Для этого станция старается передавать каждый кадр так, чтобы снизить вероятность его столкновения с другим кадром. Кроме того, каждый переданный кадр должен подтверждаться кадром квитанции, посыпаемым станцией назначения. Если же по истечении оговоренного тайм-аута квитанция не поступает, станция-отправитель считает, что произошла коллизия, и пытается передать кадр повторно.

В режиме доступа DCF делается попытка разнести во времени моменты передачи кадров станциями сети.

Для этого каждая станция, которая хочет передать кадр, обязана предварительно прослушать среду (рис. 2.32). Как только станция фиксирует окончание передачи кадра, она обязана отсчитать интервал времени, равный межкадровому интервалу (InterFrame Space, IFS). Если после истечения IFS среда все еще свободна, начинается отсчет слотов фиксированной длительности. Кадр можно начать передавать только в начале какого-либо из слотов при условии, что среда свободна. Станция выбирает слот

на основании усеченного экспоненциального двоичного алгоритма отсрочки, аналогичного тому, который используется в методе CSMA/CD. Номер слота выбирается как случайное целое число, равномерно распределенное в интервале $[0, CW]$, где CW означает **конкурентное окно** (Contention Window).

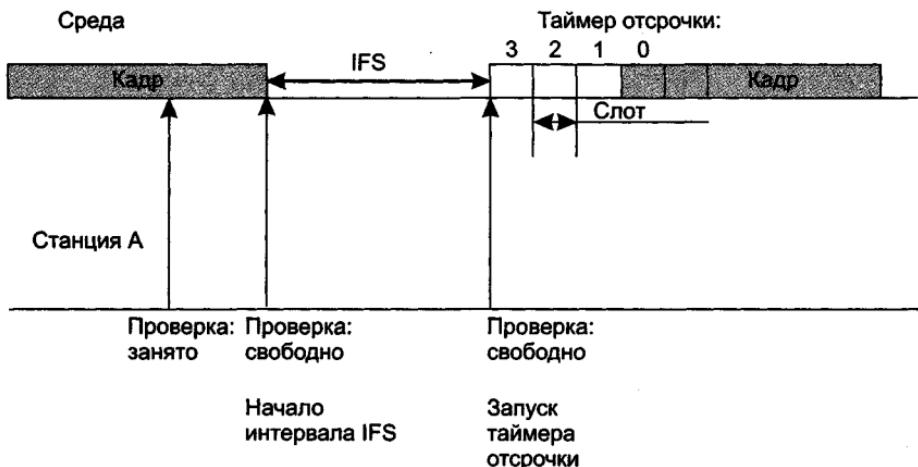


Рис. 2.32. Режим доступа DCF

Пусть станция A выбрала для передачи на основании усеченного экспоненциального двоичного алгоритма отсрочки слот 3. При этом она присваивает **таймеру отсрочки** значение 3 и начинает проверять состояние среды в начале каждого слота. Если среда свободна, то из значения таймера отсрочки вычитается 1, и если результат равен нулю, то начинается передача кадра.

Если же в начале какого-нибудь слота среда оказывается занятой, то вычитания единицы не происходит и таймер «замораживается». В этом случае станция начинает новый цикл доступа к среде. Как и в предыдущем цикле, станция следит за средой и при ее освобождении делает паузу в течение межкадрового интервала. Если среда осталась свободной, то станция *использует значение «замороженного» таймера в качестве номера слота* и выполняет описанную процедуру проверки свободных слотов с вычитанием единиц, начиная с замороженного значения таймера отсрочки. Например, если в первом цикле проверок таймер был заморожен на значении 2, то с этого значения начинается новый цикл проверки.

Размер слота выбирается таким образом, чтобы он превосходил время распространения сигнала между любыми двумя станциями сети плюс время, затрачиваемое станцией на распознавание занятости среды. Если такое условие соблюдается, то каждая станция сети сумеет правильно распознать начало передачи кадра при прослушивании слотов, предшествующих выбранному ею для передачи слоту. Это, в свою очередь, означает следующее.

Коллизия может случиться только в том случае, когда несколько станций выбирают один и тот же слот для передачи.

В этом случае кадры искажаются и квитанции от станций назначения не приходят. Не получив в течение определенного времени квитанцию, отправители фиксируют факт коллизии и пытаются передать свои кадры снова. При каждой повторной неудачной попытке передачи кадра интервал $[0, CW]$, из которого выбирается номер слота, удваивается, а при достижении некоторого предела попытки передать кадр прекращаются.

Централизованный режим доступа (Point Coordination Function, PCF) отличается от распределенного тем, что в нем точка доступа сети исполняет роль арбитра и последовательно предоставляет право доступа к среде тем станциям, которые хотят передать приоритетные (чувствительные к задержкам) кадры, например, кадры синхронного голосового трафика. Для того чтобы какая-то доля среды всегда доставалась асинхронному трафику, длительность периода захвата среды точкой доступа для поддержки режима PCF ограничена.

Персональные сети Bluetooth

Стандарт **Bluetooth** разработан группой Bluetooth SIG (Bluetooth Special Interest Group), которая была организована по инициативе компании Ericsson. Стандарт Bluetooth также адаптирован рабочей группой IEEE 802.15.1 в соответствии с общей структурой стандартов IEEE 802.

Персональные сети (Personal Area Networks, PAN) предназначены для взаимодействия устройств, принадлежащих одному владельцу, на небольшом расстоянии, обычно в радиусе 10 м. Такими устройствами могут быть ноутбук, мобильный телефон, принтер, персональный электронный помощник (Personal Digital Assistant, PDA), телевизор, а также многочисленные бытовые приборы, например холодильник, стиральная машина, нагревательный прибор.

Персональные сети должны обеспечивать как фиксированный доступ, например, в пределах дома, так и мобильный, когда владелец устройств перемещается вместе с ними между помещениями или городами.

Персональные сети во многом похожи на локальные, но у них есть и свои особенности.

- ❑ *Многие из устройств, которые могут входить в персональную сеть, гораздо проще, чем традиционный узел локальной сети — компьютер.* Кроме того, такие устройства обычно имеют небольшие габариты и стоимость. Поэтому стандарты PAN должны учитывать, что их реализация

призвана приводить к недорогим решениям, потребляющим небольшую энергию.

- ❑ *Область покрытия PAN меньше области покрытия LAN*, для взаимодействия узлов PAN часто достаточно нескольких метров.
- ❑ *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 устройства PDA в свое 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.
- ❑ *Персональные сети в гораздо большей степени, чем локальные, тяготеют к беспроводным решениям*, поскольку при соединении между собой малогабаритных и к тому же перемещаемых устройств желание избавиться от кабелей проявляется гораздо сильнее, чем при соединении компьютера с принтером или концентратором.
- ❑ *Устройство PAN должно излучать сигналы небольшой мощности*, желательно не более 100 мВт (обычный сотовый телефон излучает сигналы мощностью от 600 мВт до 3 Вт), поскольку обычно человек носит устройство PAN с собой и на себе и при этом оно не должно причинять вред его здоровью.

Сегодня самой популярной технологией PAN является **Bluetooth**, которая обеспечивает взаимодействие 8 устройств в разделяемой среде диапазона 2,4 ГГц со скоростью передачи данных до 723 Кбит/с.

В технологии Bluetooth используется концепция **пикосети**. Название подчеркивает небольшую область покрытия, от 10 до 100 м, в зависимости от мощности излучения передатчика устройства. В пикосеть может входить до 255 устройств, но только 8 из них могут в каждый момент времени быть активными и обмениваться данными. Одно из устройств в пикосети является **главным**, остальные — **подчиненными** (рис. 2.33).

Активное подчиненное устройство может обмениваться данными только с главным устройством, прямой обмен между подчиненными устройствами невозможен. Все подчиненные устройства данной пикосети, кроме семи активных, 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*, которой являются нелицензируемые частоты диапазона 2,4 ГГц. Разделяемая среда передает данные со скоростью 1 Мбит/с, но из-за накладных расходов

на заголовки пакетов и смену частот полезная скорость передачи данных в среде не превышает 777 Кбит/с. Пропускная способность среды делится главным устройством между семью подчиненными устройствами на основе техники TDM.

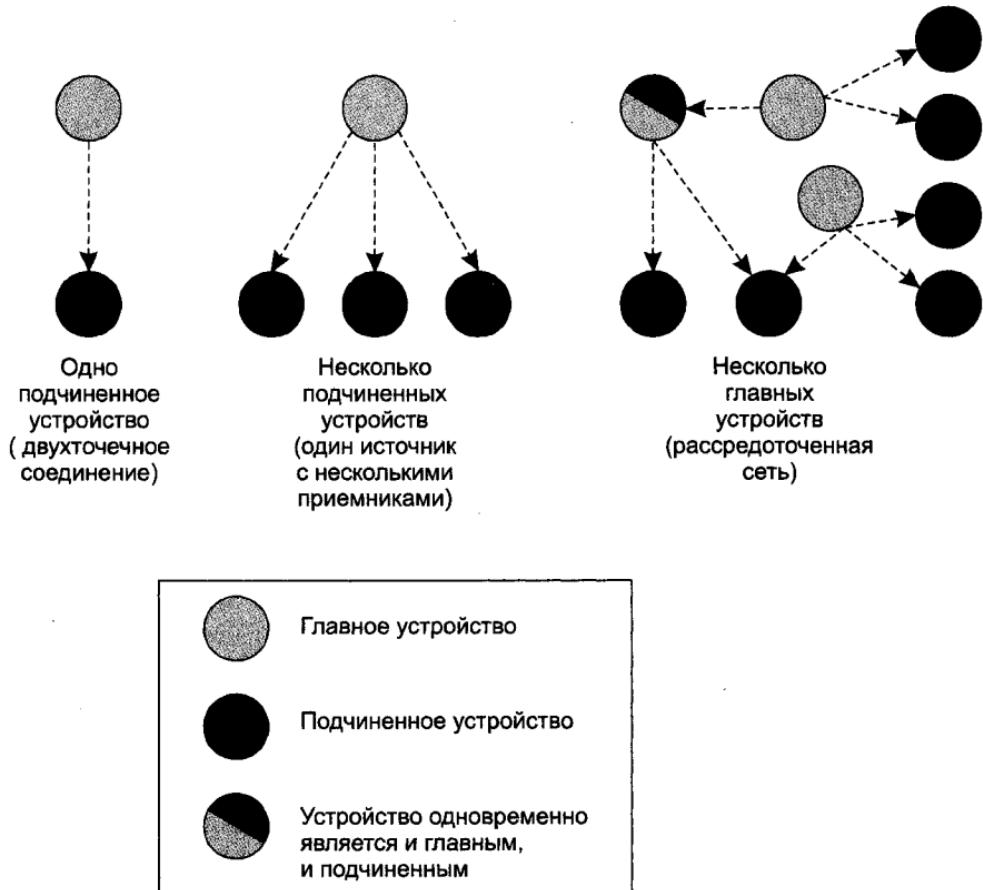


Рис. 2.33. Пикосеть и рассредоточенная сеть

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 этой сети.

Присоединение к пикосети происходит динамически. Главное устройство пикосети, используя процедуру опроса, собирает информацию об устройствах, которые попадают в зону его пикосети. После обнаружения нового устройства главное устройство проводит с ним переговоры. Если желание подчиненного устройства присоединиться к пикосети совпадает с решением главного устройства (причем подчиненное устройство прошло проверку

аутентичности и оказалось в списке разрешенных устройств), то новое подчиненное устройство присоединяется к пикосети.

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.

Для того чтобы сигналы разных пикосетей не интерферировали, каждое главное устройство задействует *собственную* последовательность псевдослучайной перестройки частоты. Использование различающихся последовательностей псевдослучайной перестройки частоты затрудняет общение пикосетей между собой. Для преодоления этой проблемы устройство, играющее роль моста, должно при подключении к каждой из пикосетей соответствующим образом менять частоту.

Коллизии, хотя и с очень небольшой вероятностью, все же могут происходить, когда два или более устройства из разных пикосетей выбирают для работы один и тот же частотный канал.

Сети Bluetooth разными методами обеспечивают передачу информации двух типов.

- ❑ Для *чувствительного к задержкам трафика* (например, голоса) сеть поддерживает **синхронный канал, ориентированный на соединение** (Synchronous Connection-Oriented link, SCO). Этот канал работает со скоростью 64 Кбит/с и для него 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 этого канала пропускная способность выделяется по запросу подчиненного устройства или по потребности главного устройства.

ГЛАВА 3. Сети TCP/IP

Сейчас мы работаем над стандартизацией протоколов [TCP/IP], которые позволяют колонизаторам планет и кораблям, путешествующим по Солнечной системе, общаться друг с другом, а также пользоваться Интернетом.

*Винт Серф,
один из ведущих разработчиков
стека TCP/IP*

В этой главе мы рассмотрим протоколы самой популярной сетевой технологии TCP/IP, которая появилась уже почти 40 лет назад как результат создания Интернета, а сегодня используется практически во всех существующих и вновь создаваемых локальных и глобальных сетях.

Мы изучим средства адресации узлов в сетях TCP/IP, включая протокол ARP и систему DNS, технологию DHCP и протокол NAT. Узнаем, как на основе протокола IP происходит объединение нескольких сетей в единую сеть, как осуществляется маршрутизация и какую роль играет в работе в такой составной сети протокол ICMP. Мы обсудим, как обеспечивает надежность доставки данных протокол транспортного уровня TCP и как он может влиять на загрузку сети.

Однако главная наша цель состоит в том, чтобы показать взаимодействие всех упомянутых протоколов. Чтобы, подобно тому, как из отдельных кусочков складывается мозаика, из описаний работы отдельных протоколов у читателя сложилась цельная картина функционирования сети TCP/IP.

Стек протоколов TCP/IP

Сегодня стек TCP/IP широко используется как в глобальных, так и локальных сетях. Этот стек имеет иерархическую структуру, в которой определено 4 уровня (рис. 3.1).

Прикладной уровень стека TCP/IP соответствует трем верхним уровням модели OSI: прикладному, представления и сеансовому. Он объединяет службы, предоставляемые системой пользовательским приложениям. За долгие годы применения в сетях различных стран и организаций стек TCP/IP накопил

большое количество протоколов и служб прикладного уровня. К ним относятся такие распространенные протоколы, как протокол передачи файлов (File Transfer Protocol, FTP), протокол эмуляции терминала telnet, простой протокол передачи электронной почты (Simple Mail Transfer Protocol, SMTP), протокол передачи гипертекста (Hypertext Transfer Protocol, HTTP) и многие другие. Протоколы прикладного уровня развертываются на хостах¹.

Прикладной уровень	FTP, Telnet, HTTP, SMTP, SNTP, TFTP
Транспортный уровень	TCP, UDP
Сетевой уровень	IP, ICMP, RIP, OSPF
Уровень сетевых интерфейсов	Не регламентируется

Рис. 3.1. Иерархическая структура стека TCP/IP

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).

Для того чтобы обеспечить надежную доставку данных, протокол TCP предусматривает установление логического соединения, что позволяет ему нумеровать пакеты, подтверждать их прием квитанциями, в случае потери организовывать 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

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.

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, который используется в том случае, когда задача надежного обмена данными либо вообще не ставится, либо решается средствами более высокого уровня — прикладного уровня 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, 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. Именно этот уровень, функции которого соответствуют сетевому уровню модели OSI, обеспечивает перемещение пакетов в пределах составной сети, образованной объединением нескольких подсетей. Протоколы сетевого уровня поддерживают интерфейс с вышележащим транспортным уровнем, получая от него запросы на передачу данных по составной сети, а также с нижележащим уровнем сетевых интерфейсов, о функциях которых мы расскажем далее.

Основным протоколом сетевого уровня является межсетевой протокол (Internet Protocol, IP). В его задачу входит продвижение пакета между сетями — от одного маршрутизатора к другому до тех пор, пока пакет не попадет в сеть назначения. В отличие от протоколов прикладного и транспортного уровней, протокол IP развертывается не только на хостах, но и на всех маршрутизаторах (шлюзах). Протокол IP — это дейтаграммный протокол, работающий без установления соединений по принципу доставки с максимальными усилиями (best effort). Такой тип сетевого сервиса называют также «ненадежным».

К сетевому уровню TCP/IP часто относят протоколы, выполняющие вспомогательные функции по отношению к IP. Это, прежде всего, протоколы маршрутизации RIP и OSPF, предназначенные для изучения топологии сети, определения маршрутов и составления таблиц маршрутизации, на основании которых протокол IP перемещает пакеты в нужном направлении. По этой же причине к сетевому уровню могут быть отнесены протокол межсетевых управляющих сообщений (Internet Control Message Protocol, ICMP), предназначенный для передачи маршрутизатором источнику сведений об ошибках, возникших при передаче пакета, и некоторые другие протоколы.

Идеологическим отличием архитектуры стека TCP/IP от многоуровневой архитектуры других стеков является интерпретация функций самого нижнего уровня — **уровня сетевых интерфейсов**.

Напомним, что нижние уровни модели OSI (канальный и физический) реализуют множество функций доступа к среде передачи, формированию кадров, согласованию величин электрических сигналов, кодированию и синхронизации, а также некоторые другие. Все эти весьма конкретные функции составляют суть таких протоколов обмена данными, как Ethernet, PPP и многих других.

У нижнего уровня стека TCP/IP задача существенно проще — он отвечает только за организацию взаимодействия с технологиями сетей, входящих в составную сеть. TCP/IP рассматривает любую подсеть, входящую в составную сеть, как средство транспортировки пакетов между двумя соседними маршрутизаторами.

Задачу обеспечения интерфейса между технологией TCP/IP и любой другой технологией промежуточной сети упрощено можно свести к определению способа:

- упаковки (инкапсуляции) IP-пакета в единицу передаваемых данных промежуточной сети;
- преобразования сетевых адресов в адреса технологии данной промежуточной сети.

Такой гибкий подход упрощает задачу 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 (например, спецификация RFC 1577, определяющая работу протокола IP через сети ATM, появилась в 1994 году вскоре после принятия основных стандартов ATM). Так как для каждой вновь появляющейся технологии разрабатываются собственные интерфейсные средства, функции этого уровня нельзя определить раз и навсегда, и именно поэтому нижний уровень стека TCP/IP не регламентируется.

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. 3.2).

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** или **датаграммой**. Дейтаграмма — это общее название для единиц данных, которыми

оперируют протоколы без установления соединений. К таким протоколам относится и протокол IP, поэтому его единицу данных также называют дейтаграммой. Однако очень часто используется и другой термин — **пакет**.

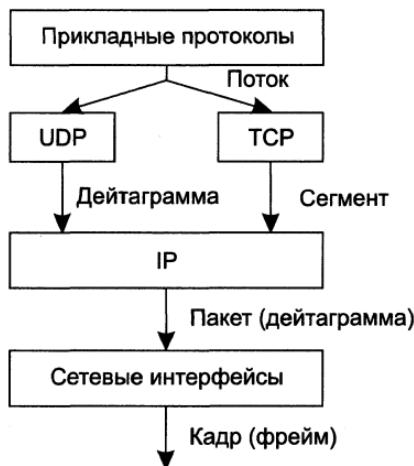


Рис. 3.2. Названия протокольных единиц данных в TCP/IP

В стеке TCP/IP единицы данных любых технологий, в которые упаковываются IP-пакеты для последующей переноски их через сети составной сети, принято называть также **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 ячейка ATM, и пакет X.25 в тех случаях, когда они выступают в качестве контейнера, в котором IP-пакет передается через составную сеть.

Адресация в сетях TCP/IP

Одним из важных 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.

Типы адресов стека TCP/IP

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, которые предназначены исключительно для обеспечения связи собственных узлов. Однако как только некоторая сеть объединяется

с другими сетями в составную сеть, функциональность этих адресов расширяется, они становятся необходимым элементом вышеследящей объединяющей технологии — в данном случае технологии 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 — 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 как LAN, так и WAN. Следовательно, говоря о подсетеи, 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 подсетеи 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 узла¹, 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 (доменными) именами** компьютеров. Символьные имена в пределах составной сети строятся по иерархическому признаку. Примером доменного имени может служить имя base2.sales.zil.ru. Символьные имена называют также **DNS-именами**.

¹ Напомним, что для сокращения мы часто говорим «узел» вместо «сетевой интерфейс узла».

Между локальным адресом, доменным именем и IP-адресом, относящимся к одному и тому же сетевому интерфейсу, нет никакой функциональной зависимости. В общем случае сетевой интерфейс может иметь несколько локальных адресов, сетевых адресов, доменных имен.

Формат IP-адреса

В заголовке IP-пакета для хранения IP-адресов отправителя и получателя отводятся два поля, каждое имеет фиксированную длину 4 байт (32 бит). IP-адрес состоит из двух логических частей — номера сети и номера узла в сети.

Наиболее распространенной формой представления IP-адреса является запись в виде четырех чисел, представляющих значения каждого байта в десятичной форме и разделенных точками, например:

128.10.2.30

В некоторых случаях оказывается полезным представление IP-адреса в двоичном формате:

10000000 00001010 00000010 00011110.

Заметим, что запись адреса *не предусматривает специального разграничитывающего знака* между номером сети и номером узла. Вместе с тем при передаче пакета по сети часто возникает необходимость автоматическими средствами разделить адрес на эти две части. Например, маршрутизация, как правило, осуществляется на основании номера сети, поэтому каждый маршрутизатор, получив пакет, должен прочитать из соответствующего заголовка адрес назначения и выделить из него номер сети. Таким образом маршрутизаторы определяют, какая часть из 32 бит, отведенных под IP-адрес, относится к номеру сети, а какая — к номеру узла?

Для этих целей служат два подхода.

- Первый способ (RFC 950, RFC 1518) основан на использовании маски, которая позволяет максимально гибко устанавливать границу между номером сети и номером узла. **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. Например, если маска, связываемая с некоторым IP-адресом, имеет вид 11111111110000000000000000000000, то номеру сети соответствуют 10 старших разрядов в двоичном представлении данного IP-адреса.
- Второй наиболее распространенный до недавнего времени способ решения данной проблемы заключается в использовании **классов адресов** (RFC 791). Вводится пять классов адресов: A, B, C, D, E. Три из них — A, B и C — служат 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

Признаком, на основании которого IP-адрес относится к тому или иному классу, являются значения нескольких первых битов адреса. Рисунок 3.3 иллюстрирует структуру IP-адресов разных классов.



Рис. 3.3. Классы IP-адресов

Исходя из приведенной структуры адресов и информации из табл. 3.1, можно сделать несколько очевидных выводов. Сетей класса А сравнительно немного, зато количество узлов в них очень большое, достигает значения 2^{24} , что равно 16 777 216 узлов. Сетей класса В больше, чем сетей класса А, но их размеры меньше, максимальное количество узлов в сетях класса В составляет 2^{16} (65 536). Сетей класса С больше всего, но они характеризуются самым маленьким максимальным количеством узлов — всего 2^8 (256).

В то время как адреса классов А, В и С используются для идентификации отдельных сетевых интерфейсов, то есть являются **индивидуальными адресами** (unicast address), адреса класса D являются **групповыми** (multicast address) и идентифицируют группу сетевых интерфейсов, которые в общем случае могут принадлежать разным сетям. Интерфейс, входящий в группу, получает наряду с обычным индивидуальным IP-адресом еще один групповой адрес. Если при отправке пакета в качестве адреса назначения указан адрес класса D, то такой пакет должен быть доставлен всем узлам, которые входят в группу.

В табл. 3.1 приведены диапазоны адресов и максимальное количество сетей и узлов, соответствующих каждому классу.

Таблица 3.1. Характеристики адресов разного класса

Класс адресов	Первые биты	Диапазон адресов в пределах класса	Особые адреса	Максимальное количество сетей и узлов в сети класса
A	0	0.0.0.0 – 127.255.255.255	0.0.0.0 – не используется, 127.0.0.0 – зарезервирован	Сетей 2^7 , узлов 2^{24}
B	10	128.0.0.0 – 191.255.255.255		Сетей 2^{15} , узлов 2^{16}
C	110	192.0.0.0 – 223.255.255.255		Сетей 2^{21} , узлов 2^8
D	1110	224.0.0.0 – 239.255.255.255	Групповые адреса	Количество адресов групп 2^{28}
E	11110	240.0.0.0 – 247.255.255.255	Зарезервированы	Количество зарезервированных адресов 2^{27}

Чтобы получить из IP-адреса номер сети и номер узла, требуется не только разделить адрес на две части, соответствующие номерам сети и узла, но и дополнить каждую из них нулями до полных четырех байтов. Возьмем, например, адрес 129.64.134.5 класса В. Первые два байта идентифицируют сеть, а последующие два — узел. Таким образом, номером сети является 129.64.0.0, а номером узла — 0.0.134.5.

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. 3.1 для сетей каждого класса, должно быть уменьшено на 2. Например, в адресах класса С под номер узла отводится 8 бит, которые позволяют задавать 256 номеров: от 0 до 255. Однако в действительности максимальное количество узлов в сети класса С не может превышать 254, так как адреса 0 (двоичное представление — 00000000) и 255 (11111111) 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 А 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 рассыпаться всем узлам, находящимся в той же сети, что и источник этого 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, рассыпается *всем* узлам сети, номер которой указан в адресе назначения. Например, пакет с адресом 192.190.21.255 будет направлен всем 254 узлам сети 192.190.21.0. Такой тип адреса называется **широковещательным** (*broadcast*).

ПРИМЕЧАНИЕ

В протоколе IP нет понятия широковещания 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 вариант широковещательной рассылки имеет пределы распространения в составной сети — они ограничены либо сетью, содержащей источник пакета, либо сетью, номер которой указан в адресе назначения. Поэтому деление сети с помощью маршрутизаторов на части локализует широковещательный штурм 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 служит 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. Когда программа посылает данные по IP-адресу 127.x.x.x, то данные не передаются в сеть, а возвращаются модулям верхнего уровня того же компьютера, как если бы они были приняты из сети. Маршрут перемещения данных образует «петлю», поэтому этот адрес называется **адресом обратной петли** (*loopback*).

Групповые адреса (*multicast*), относящиеся к классу D, 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. В общем случае члены группы могут распределяться по различным сетям, находящимся друг от друга на произвольно большом расстоянии. Групповой адрес не делится на номера сети и узла и обрабатывается маршрутизатором особым образом. Основное назначение групповых адресов — распространение информации по схеме «один ко многим». От того, найдут ли разработчики сетевых средств эффективный способ использования групповых адресов (пока они применяются лишь в небольших экспериментальных «островках» в Интернете), зависит, сможет ли Интернет создать серьезную конкуренцию радио и телевидению.

Использование масок при IP-адресации

Снабжая каждый IP-адрес маской, можно отказаться от понятий классов адресов и сделать более гибкой систему адресации.

Пусть, например, для IP-адреса 129.64.134.5 указана маска 255.255.128.0, то есть в двоичном виде IP-адрес 129.64.134.5 равен:

10000001.01000000.10000110.00000101,

в то время как маска 255.255.128.0 выглядит так:

1111111.1111111.1000000.00000000.

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 В).

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.

Для записи масок используются и другие форматы. Например, удобно интерпретировать значение маски, записанной в шестнадцатеричном коде: FF.FF.00.00 — маска для адресов класса В. Еще чаще встречается обозначение 185.23.44.206/16 — эта запись сообщает, что адрес имеет значение 185.23.44.206 и что в указанном IP-адресе под номер сети отведено 16 двоичных разрядов.

Механизм масок широко распространен в сфере IP-маршрутизации, причем маски могут использоваться для самых разных целей. С их помощью администратор может разбить одну выделенную ему поставщиком услуг

сеть определенного класса на несколько других, не требуя от него дополнительных номеров сетей, — эта операция называется *разделением на подсети* (subnetting). На основе этого же механизма поставщики услуг могут объединять адресные пространства нескольких сетей путем введения так называемых « префиксов» с целью уменьшения объема таблиц маршрутизации и повышения за счет этого производительности маршрутизаторов — такая операция называется *объединением подсетей* (supernetting). Подробнее об этом мы поговорим при изучении технологии бесклассовой междоменной маршрутизации.

В табл. 3.2 помещены значения масок для стандартных классов сетей.

Таблица 3.2. 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	FFFF.FF.00	/24

Порядок назначения IP-адресов и технология CIDR

По определению схема IP-адресации должна обеспечивать уникальность нумерации сетей, а также узлов в пределах каждой из сетей. Когда дело касается сети, являющейся частью Интернета, уникальность нумерации в пределах всей этой огромной сети может быть обеспечена только усилиями специально созданных для этого центральных органов. В небольшой же автономной IP-сети условие уникальности номеров сетей и узлов может быть выполнено силами сетевого администратора.

В этом случае в распоряжении администратора имеется все адресное пространство, 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. Однако при таком подходе исключена возможность в будущем подсоединить данную сеть к Интернету. Действительно, произвольно выбранные адреса данной сети могут совпасть с централизованно назначенными адресами Интернета. Для того 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 использование частных адресов для адресации автономных сетей делает возможным корректное подключение их к Интернету. Применяемые при этом специальные технологии подключения¹ исключают коллизии адресов.

В больших сетях, подобных Интернету, уникальность сетевых адресов гарантируется централизованной иерархически организованной системой их распределения. Главным органом регистрации глобальных адресов в Интернете с 1998 г. 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. При этом надо отметить, что дефицит обусловлен не только ростом количества сетей и их размеров, но и тем, что имеющееся адресное пространство используется нерационально. Действительно, размеры сетей, относящихся к разным классам, резко различаются, например, клиент, получивший сеть класса А, становится обладателем 16 777 216 индивидуальных адресов, класса В – 65 536, а класса С – 256. Как видим, эта градация слишком грубая, что во многих случаях не позволяет центрам распределения адресов избежать выдачи абонентам излишних адресов.

Принципиальным решением этой проблемы является переход на новую версию протокола IP – протокол IPv6, в котором резко расширяется адресное пространство. Однако и текущая версия протокола IP (IPv4) поддерживает технологии, направленные на более экономное расходование IP-адресов, такие, например, как CIDR.

Технология бесклассовой междоменной маршрутизации (Classless Inter-Domain Routing, CIDR)² основана на использовании масок для более гибкого распределения адресов и более эффективной маршрутизации. Она допускает произвольное разделение IP-адреса на поле для номера сети и поле для номеров узлов. При такой системе адресации клиенту может быть выдан пул адресов, более точно соответствующий его запросу, чем это происходит при адресации на основе классов адресов. Например, если клиенту А (рис. 3.4)

¹ Например, такой технологией является NAT, которая будет рассмотрена далее.

² Технология CIDR описана в документах RFC 1517, RFC 1518, RFC 1519, RFC 1520.

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, имеющая вид *IP-адрес/маска*, 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».

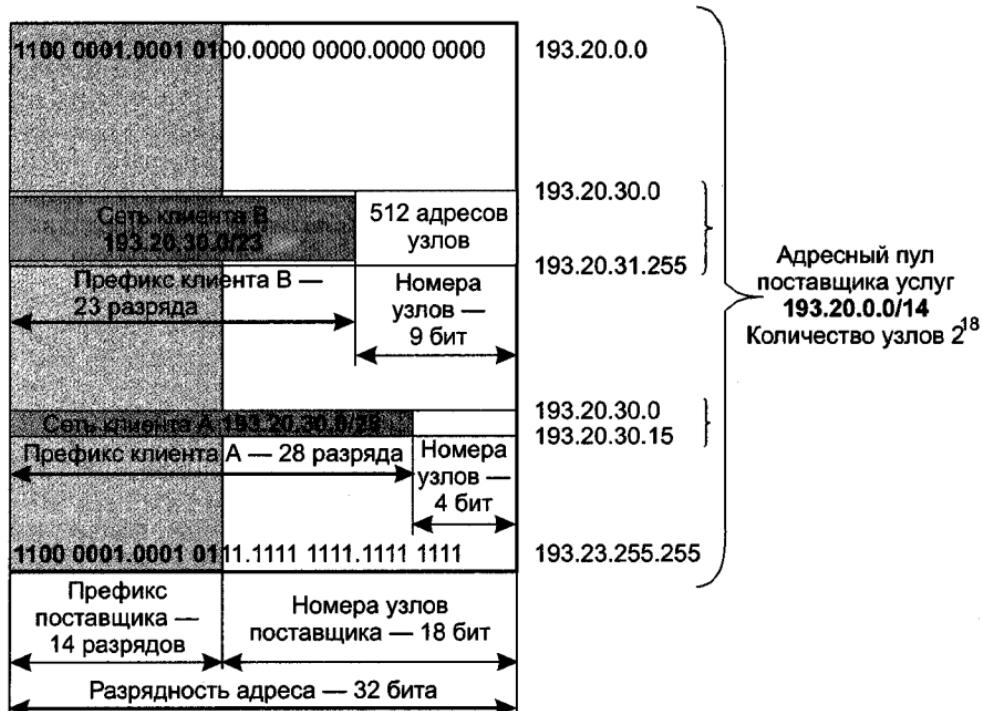


Рис. 3.4. Схема распределения адресного пространства в технологии CIDR

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 *непрерывным*. При таком условии все адреса имеют общий **префикс** — одинаковую последовательность цифр в старших разрядах адреса.

Пусть, например, как показано на рис. 3.4, провайдер располагает адресами в диапазоне 193.20.0.0–193.23.255.255 (или в двоичной записи 1100 0001.0001 0110.0000 0000–1100 0001.0001 0111.1111 1111.1111 1111). Здесь префикс провайдера имеет длину 14 разрядов (1100 0001.0001 01), что можно

записать в виде 193.20.0.0/14. Префикс обычно интерпретируется как номер подсети.

Даже если необходимое клиенту адресное пространство может быть предоставлено с помощью нескольких сетей стандартного класса, предпочтительным считается вариант IP-адрес/маска, так как в этом случае адреса гарантированно образуют непрерывное пространство. Непрерывность адресного пространства является очень важным свойством, непосредственно влияющим на эффективность маршрутизации, о чём мы будем говорить далее при изучении маршрутизации с использованием масок.

Рассмотрим еще один пример. Пусть клиент В собирается связать в сеть 500 компьютеров (см. рис. 3.4). Вместо того чтобы выделять ему две сети класса С по 256 узлов каждая, клиенту назначают пул адресов в виде пары 193.20.30.0/23. Эта запись означает, что клиенту выделена сеть неопределенного класса, в которой под нумерацию узлов отведено 9 младших битов, что, как и в случае двух сетей класса С, позволяет адресовать 512 узлов. Преимущество этого варианта с маской перед вариантом с двумя сетями состоит в том, что, в первом случае *непрерывность пула адресов гарантирована*.

Назначение адресов в виде IP-адрес/маска корректно, только если поле для адресации узлов, полученное применением маски к IP-адресу, содержит только одни нули. Например, определение пула адресов в виде 193.20.00.0/12 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/ 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 адрес должен состоять только из нулей;
- количество адресов в пуле равно $2^{(32 - n)}$.

Благодаря CIDR поставщик услуг получает возможность «нарезать» блоки из выделенного ему адресного пространства в соответствии с действительными требованиями каждого клиента.

Протокол ARP

Как уже было сказано, 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 представить как таблицу, состоящую из одной строки. Проблема состоит в том, как организовать обмен имеющейся информацией между узлами сети.

Для определения локального адреса по IP-адресу используется **протокол разрешения адресов** (Address Resolution Protocol, ARP). Протокол разрешения адресов реализуется различным образом в зависимости от того, работает ли в данной сети протокол локальной сети с возможностью широковещания (Ethernet) или же какой-либо из протоколов глобальной сети (ATM, Frame Relay), которые не поддерживают широковещательный доступ.

Рассмотрим работу протокола ARP в локальных сетях с широковещанием.

Протокол ARP поддерживает на каждом интерфейсе сетевого адаптера или маршрутизатора 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.

На рис. 3.5 показан фрагмент IP-сети, включающий две сети — Ethernet 1 и Ethernet 2, подключенные к интерфейсам 1 и 2 маршрутизатора соответственно.

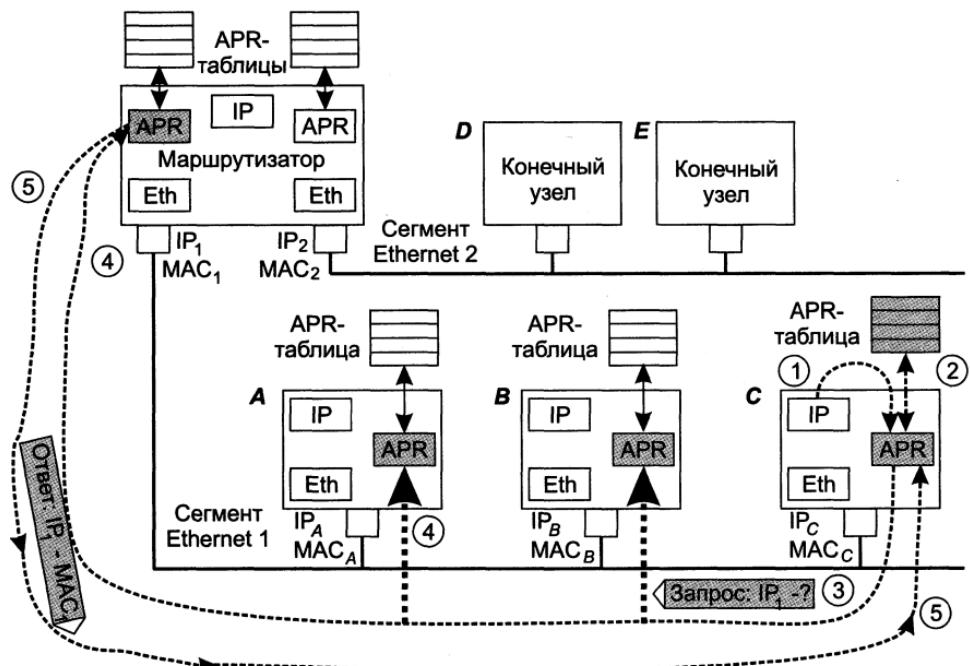


Рис. 3.5. Схема работы протокола ARP

Пусть в какой-то момент IP-модуль узла *C* направляет пакет узлу *D*. Протокол IP узла *C* в результате конфигурирования стал известен IP-адрес

интерфейса следующего маршрутизатора – это IP₁. Однако для того, чтобы направить пакет маршрутизатору, необходимо определить его локальный адрес (MAC-адрес). Для решения этой задачи предпринимаются следующие шаги:

1. На первом шаге происходит передача от протокола IP протоколу ARP примерно такого сообщения: «Какой MAC-адрес имеет интерфейс с адресом IP₁?»
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 рассыпает. Заметим, что зона распространения ARP-запроса ограничивается сетью Ethernet 1, так как на пути широковещательных кадров барьером стоит маршрутизатор.
4. Все интерфейсы сети Ethernet 1 получают ARP-запрос и направляют его «своему» протоколу ARP. ARP сравнивает указанный в запросе адрес IP₁ с IP-адресом собственного интерфейса.
5. Протокол ARP, который констатировал совпадение (в данном случае это ARP интерфейса 1 маршрутизатора), формирует ARP-ответ. В ARP-ответе маршрутизатор указывает локальный адрес MAC₁, соответствующий адресу IP₁ своего интерфейса, и отправляет его запрашивающему узлу (в данном примере узлу C).

Чтобы уменьшить число ARP-обращений в сети, найденное соответствие между IP-адресом и MAC-адресом запоминается в ARP-таблице компьютера C (в данном случае это запись: IP₁ – MAC₁). Теперь, если вдруг вновь возникнет необходимость послать пакет по адресу IP₁, соответствующий локальный адрес будет быстро извлечен из ARP-таблицы.

ARP-таблица пополняется *не только за счет поступающих на данный интерфейс ARP-ответов*, но и в результате извлечения полезной информации из широковещательных ARP-запросов. Поскольку в каждом запросе 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 нашем примере все узлы, получившие ARP-запрос от узла C, могут пополнить свои ARP-таблицы записью: IP_C – MAC_C.

В ARP-таблицах существует два типа записей: динамические и статические. **Статические записи** создаются вручную с помощью утилиты арг 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**.

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-таблицы, в которых он задает, например, соответствие IP-адресов номерам виртуальных каналов, имеющих 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, который ведет ARP-таблицу для всех остальных узлов и маршрутизаторов этой сети. Этот маршрутизатор называют **ARP-сервером**. Единственное, что требуется сделать вручную — это занести в память всех компьютеров и маршрутизаторов сети IP-адрес и локальный адрес ARP-сервера. При включении каждый узел и маршрутизатор регистрирует свои адреса на 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.

Доменные имена

В стеке TCP/IP применяется доменная система имен, которая имеет иерархическую древовидную структуру, допускающую наличие в имени произвольного количества составных частей (рис. 3.6).

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 — хосту. 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. 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.

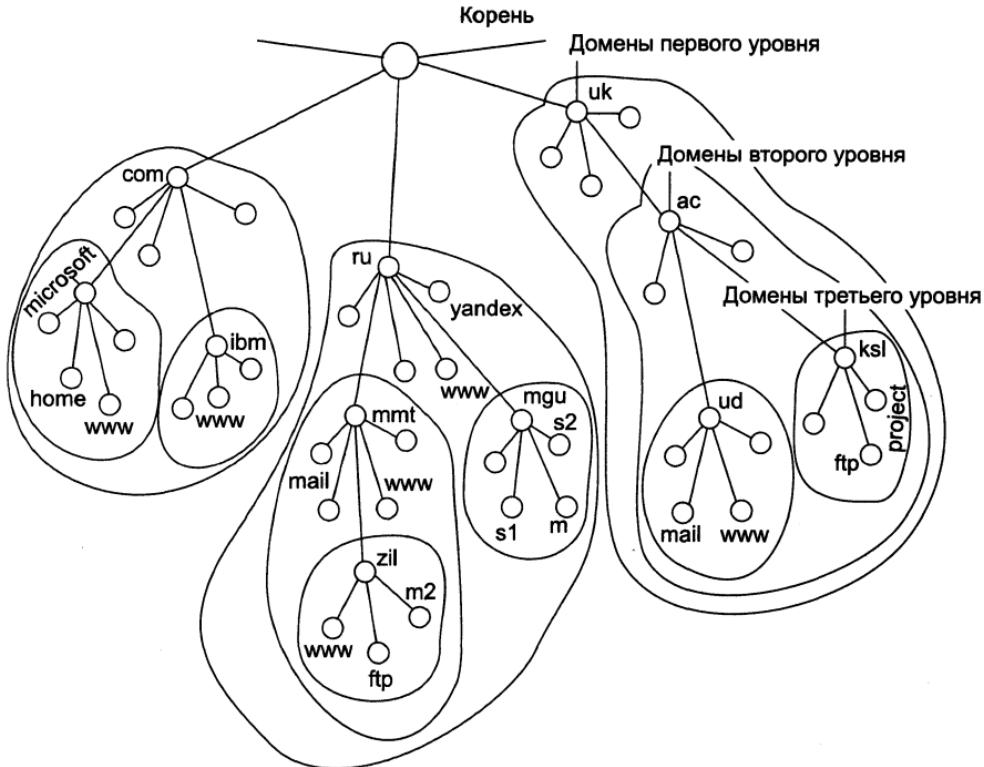


Рис. 3.6. Пространство доменных имен

Совокупность имен, у которых несколько старших составных частей совпадают, образуют **домен имен** (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 — `ru`. Другим примером является домен `mgu.ru`. Из представленных на рис. 3.6 имен в него входят имена `s1.mgu.ru`, `s2.mgu.ru` и `m.mgu.ru`. Этот домен образуют имена, у которых две старшие части всегда 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.ru`, так как имеют общую старшую часть имени. Часто поддомены для краткости называют только младшей частью имени, то есть поддомены `s1`, `s2` и `m`.

ВНИМАНИЕ

Компьютеры, имена которых относятся к одному и тому же домену, могут иметь абсолютно независимые друг от друга IP-адреса, принадлежащие различным 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 управляетя центральными органами Интернета, в частности уже упоминавшейся нами организацией ICANN. Домены верхнего уровня назначаются для каждой страны, а также для различных типов организаций. Имена этих доменов должны следовать международному стандарту ISO 3166. Для обозначения стран используются трехбуквенные и двухбуквенные аббревиатуры, например, ru (Россия), uk (Великобритания), fi (Финляндия), us (Соединенные Штаты), а для различных типов организаций могут применяться, в частности, 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 верхнего уровня администрирует отдельная организация, уполномоченная центральными органами Интернета, которая обычно разбивает свой домен на поддомены и передает функции администрирования этих поддоменов другим организациям, распределяющим доменные имена среди своих клиентов.

Система DNS

Широковещательный механизм установления соответствия между символьными именами и локальными адресами, подобный протоколу ARP, хорошо работает только в небольшой локальной сети, не разделенной на подсети. В крупных сетях, где возможность всеобщей широковещательной рассылки не поддерживается, нужен другой инструмент разрешения символьных имен.

На раннем этапе развития Интернета на каждом хосте вручную создавался текстовый файл 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, поддерживать их становилось все сложнее, и создание *масштабируемого* решения для разрешения имен стало необходимостью.

Таким решением стала **система доменных имен** (Domain Name System, DNS). Эта служба состоит из некоторого количества серверов, рассеянных по всей составной сети, и множества клиентов, работающих практически на каждом конечном узле. DNS-серверы поддерживают распределенную базу отображений «доменное имя — IP-адрес», а DNS-клиенты обращаются к серверам с запросами о разрешении доменного имени в IP-адрес.

Служба DNS использует текстовые файлы почти такого формата, как и файлы hosts.txt, и эти файлы администратор также подготавливает вручную.

Однако в базе данных DNS предусмотрено несколько типов записей. Помимо основного типа записей — A, в которых устанавливается соответствие между DNS-именами и IP-адресами хостов, имеются и другие типы записей, расширяющие функциональность системы DNS. Так, например, запись типа MX указывает DNS-имя почтового сервера, относящегося к тому или иному домену имен, а запись типа SOA содержит электронный адрес и другую идентифицирующую информацию об администраторе, который создавал записи для этой базы данных.

Система DNS является распределенной, она опирается на иерархию доменов, и каждый сервер службы DNS хранит только часть имен сети, а не все имена, как это происходит при использовании файлов hosts.txt. При росте количества узлов в сети проблема масштабирования решается созданием новых доменов и поддоменов имен и добавлением в службу DNS новых серверов.

Для каждого домена имен создается свой DNS-сервер. Имеется два механизма распределения имен на серверах. В первом случае сервер может хранить отображения «доменное имя — IP-адрес» для всего домена, включая все его поддомены. Однако такое решение оказывается плохо масштабируемым, так как при добавлении новых поддоменов нагрузка на этот сервер может превысить его возможности. Чаще используется другой подход, когда сервер домена хранит только имена, которые заканчиваются на следующем ниже уровне иерархии по сравнению с именем домена (аналогично каталогу файловой системы, который содержит записи о файлах и подкаталогах, непосредственно в него «входящих»). Именно при такой организации службы DNS нагрузка по разрешению имен распределяется более-менее равномерно между всеми DNS-серверами сети. Например, в первом случае DNS-сервер домена mmt.ru будет хранить отображения для всех имен, заканчивающихся суффиксом mmt.ru (www.zil.mmt.ru, ftp.zil.mmt.ru, mail.mmt.ru и т. д.). Во втором случае этот сервер хранит отображения только имен типа mail.mmt.ru, www.mmt.ru, а все остальные отображения должны храниться на DNS-сервере поддомена zil.

Каждый DNS-сервер помимо таблицы отображений имен содержит ссылки на DNS-серверы своих поддоменов. Эти ссылки связывают отдельные DNS-серверы в единую службу DNS. Ссылки представляют собой IP-адреса соответствующих серверов. Для обслуживания корневого домена выделено несколько дублирующих друг друга DNS-серверов, IP-адреса которых являются общедоступными.

Существует две основные схемы разрешения DNS-имен. В первом варианте, называемом *итеративным*, DNS-клиент сам координирует работу по поиску адреса, циклически обращаясь с запросами к разным серверам имен:

1. DNS-клиент обращается к корневому DNS-серверу с указанием полного доменного имени.
2. DNS-сервер сообщает клиенту адрес следующего DNS-сервера, обслуживающего домен верхнего уровня, заданный в следующей старшей части запрошенного имени.

3. DNS-клиент обращается к следующему DNS-серверу, который отсылает его к DNS-серверу нужного поддомена, и т. д., пока не будет найден DNS-сервер, в котором хранится соответствие запрошенного имени IP-адресу. Этот сервер дает окончательный ответ клиенту.

Во втором варианте реализуется *рекурсивная* процедура:

1. DNS-клиент обращается с запросом к локальному DNS-серверу, то есть серверу, находящемуся с ним в одном домене (либо к DNS-серверу, адрес которого указан в его конфигурационных параметрах). Запрос представляет собой сообщение определенного формата, и смысл его сводится к простому вопросу, например: «Какой IP-адрес имеет хост mail.mmt.ru?»
2. Далее возможны два варианта действий:
 - a) Если локальный DNS-сервер знает ответ, то он сразу же возвращает его клиенту (это может произойти, когда запрошенное имя входит в тот же поддомен, к которому относится DNS-клиент, или когда сервер уже узнавал данное соответствие для другого клиента и сохранил его в своем кэше).
 - b) Если локальный DNS-сервер ответа не знает, то он обращается к корневому DNS-серверу с указанием полного доменного имени. Корневой DNS-сервер сообщает локальному DNS-серверу адрес следующего DNS-сервера, обслуживающего домен верхнего уровня, заданный в следующей старшей части запрошенного имени. Локальный DNS-сервер делает запрос к следующему DNS-серверу, который отсылает его к DNS-серверу нужного поддомена и т. д., пока не будет найден DNS-сервер, в котором хранится соответствие запрошенного имени IP-адресу. Получив искомый IP-адрес, локальный DNS-сервер передает его DNS-клиенту.

Для ускорения поиска IP-адресов DNS-серверы широко применяют *кэширование* проходящих через них ответов. Чтобы служба DNS могла оперативно отрабатывать изменения, происходящие в сети, ответы кэшируются на относительно короткое время — обычно от нескольких часов до нескольких дней.

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 при выполнении процедуры конфигурирования интерфейса, для компьютера сводящейся, например, к заполнению системы экраных 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**, необходимых для его эффективной работы, например, маска и IP-адрес маршрутизатора 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 хостов¹ (Dynamic Host Configuration Protocol, DHCP) *автоматизирует процесс конфигурирования* сетевых интерфейсов, предотвращая дублирование адресов благодаря централизованному управлению их распределением. Кроме того, DHCP делает возможным *динамическое разделение адресов*, позволяя строить IP-сеть, количество узлов в которой превышает количество имеющихся в распоряжении администратора IP-адресов.

Протокол DHCP работает в соответствии с моделью *клиент-сервер*. Когда компьютер включают, установленный на нем DHCP-клиент посыпает ограниченное широковещательное сообщение, предназначенное для поиска DHCP-сервера. DHCP-сервер, который должен находиться в одной подсети с клиентами, откликается и посыпает сообщение-ответ, содержащее IP-адрес и некоторые другие конфигурационные параметры.

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 должен предварительно выполнить конфигурирование DHCP-сервера, сообщив ему один или несколько диапазонов доступных для распределения IP-адресов. Все эти адреса должны относиться к одной и той же подсети.

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 выдает определенному DHCP-клиенту один и тот же назначенный ему администратором IP-адрес (а также набор других конфигурационных параметров²).

В режиме *автоматического назначения статических адресов* DHCP-сервер самостоятельно, без вмешательства администратора, произвольным образом выбирает клиенту IP-адрес из пула наличных IP-адресов. Адресдается 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

¹ Работа DHCP описана в RFC 2131 и 2132.

² Для краткости иногда мы будем опускать это уточнение.

в момент первого назначения DHCP-сервером IP-адреса клиенту. При всех 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**. Срок аренды диктует, как долго компьютер может применять назначенный IP-адрес, перед тем как снова запросить его от 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 — до нескольких дней или даже недель.

Когда компьютер, являющийся DHCP-клиентом, удаляется из подсети, назначенный ему IP-адрес автоматически освобождается и может быть назначен другому DHCP-клиенту. В общем случае при каждом следующем подключении к сети компьютеру автоматически назначается новый адрес. Ни пользователь, ни сетевой администратор не вмешиваются в этот процесс. Такое свойство DHCP дает возможность динамически разделять адреса между несколькими компьютерами.

Давайте рассмотрим преимущества динамического разделения пула адресов на примере организации, в которой сотрудники 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. Однако вспомним, что сотрудники в этой организации редко появляются в офисе, значит, большая часть ресурсов при таком решении будет простаивать.

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 тут возникает другая проблема: кто и как будет конфигурировать компьютеры, состав которых постоянно меняется?

И эта проблема имеет два решения. 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, следовательно, это — плохое решение. Гораздо привлекательнее выглядят возможности *автоматического динамического назначения* адресов DHCP. Действительно, администратору достаточно один раз при настройке

DHCP-сервера указать диапазон из 64 адресов, а каждый вновь прибывающий мобильный пользователь будет просто физически подключать в сеть свой компьютер, на котором запускается DHCP-клиент. Он запросит конфигурационные параметры и 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.

Протокол межсетевого взаимодействия

Этот раздел посвящен протоколу IP (Internet Protocol – межсетевой протокол), описанному в документе RFC 751.

Протокол IP относится к дейтаграммным протоколам, то есть протоколам, работающим *без установления соединений*, он поддерживает обработку каждого IP-пакета как независимой единицы обмена, не связанной с другими IP-пакетами. Основной задачей протокола IP является доставка пакета через составную сеть. Эта задача *маршрутизации*, а также поддержания интерфейсов с протоколами выше- и нижележащего уровней. Протокол IP реализует политику доставки с максимальными усилиями, то есть 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 исправления этой ошибки. Например, если на промежуточном маршрутизаторе пакет был отброшен из-за ошибки по контрольной сумме, то модуль IP не пытается заново послать потерянный пакет.

Формат IP-пакета

Функционирование любого протокола связано с обработкой той служебной информации, которая переносится в полях заголовка пакета. Изучая назначение каждого поля заголовка IP-пакета, мы не только получаем формальные знания о структуре пакета, но и знакомимся с основными функциями протокола IP.

IP-пакет состоит из заголовка и поля данных. Поля заголовка показаны на рис. 3.7.

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 (до 60 байт).

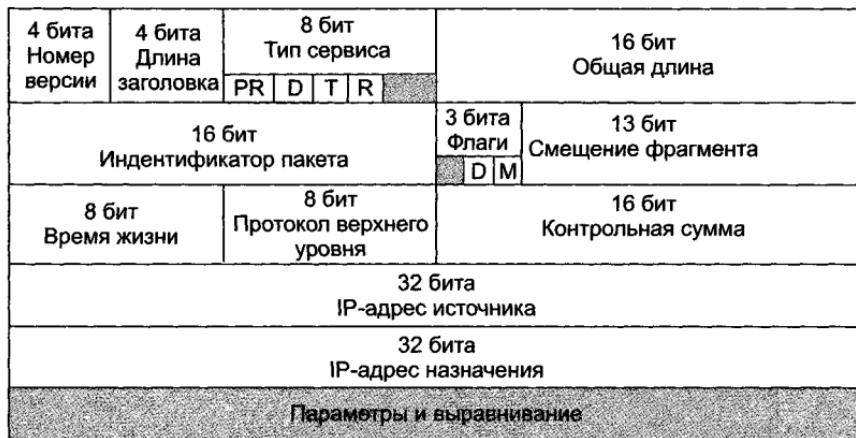


Рис. 3.7. Структура заголовка IP-пакета

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, которые отражают 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. Следующие три бита поля ToS определяют *критерий выбора маршрута*. Если бит D (Delay — задержка) установлен в 1, то маршрут должен выбираться для минимизации задержки доставки данного пакета, установленный бит T (Throughput — пропускная способность) служит для максимизации пропускной способности, а бит R (Reliability — надежность) — для максимизации надежности доставки. Оставшиеся два бита имеют нулевое значение. Стандарты дифференциированного обслуживания, принятые в конце 90-х годов, дали новое название этому полю (DS-байт) и переопределенное назначение его битов. В DS-байте также используются только старшие 6 бит, а 2 младших бита остаются в качестве резерва.

Поле **общей длины** занимает 2 байта и содержит значение общей длины пакета с учетом заголовка и поля данных. Разрядность данного поля ограничивает максимальную длину пакета величиной 65 535 байта, однако 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. Например, для передачи по сети Ethernet длина IP-пакетов не должна превышать 1500 байт.

Следующие несколько полей, включая поля **идентификатора пакета** (2 байта), **флагов MF** и **DF** (каждый по одному биту и один бит резервный),

смещения фрагмента (13 бит) и **времени жизни** (1 байт), используются при фрагментации, то есть делении пакета на части. Мы подробно поясним назначение этих полей позже при изучении фрагментации IP-пакетов.

Поле **протокола верхнего уровня** занимает один 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. Например, 6 означает, что в пакете находится сообщение TCP, 17 – сообщение UDP, 1 – сообщение 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 нуль. 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, регистрировать проходимые пакетом маршрутизаторы, помещать данные системы безопасности или временные 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.

Таблица маршрутизации

В главе 1 мы уже обсуждали принципы продвижения данных на основе таблиц коммутации (маршрутизации). Сейчас мы остановимся более детально на том, как происходит маршрутизация в IP-сетях.

Рассмотрим механизм IP-маршрутизации на примере составной сети, представленной на рис. 3.8. В этой сети шесть маршрутизаторов, $R1, R2, \dots, R6$, связывают между собой несколько подсетей и глобальную сеть Интернет.

Маршрутизаторы имеют по несколько интерфейсов (портов), к которым присоединяются сети. Каждый интерфейс маршрутизатора можно рассматривать как отдельный узел сети с сетевым и локальным адресами. Например, маршрутизатор $R2$ имеет три интерфейса, один из которых относится к сети 200.5.16.0, другой – к сети 198.21.17.0, а третий – к одной из сетей, входящих в Интернет (она отдельно не показана). Для некоторых портов

маршрутизаторов на рисунке приведены их сетевые адреса. Например, интерфейс маршрутизатора R_2 , которым он подключен к сети 198.21.17.0, имеет адрес 198.21.17.7, где 198.21.17.0 — это номер сети, а 0.0.0.4 — номер узла. Как единое устройство маршрутизатор не имеет отдельно ни сетевого, ни локального адреса.

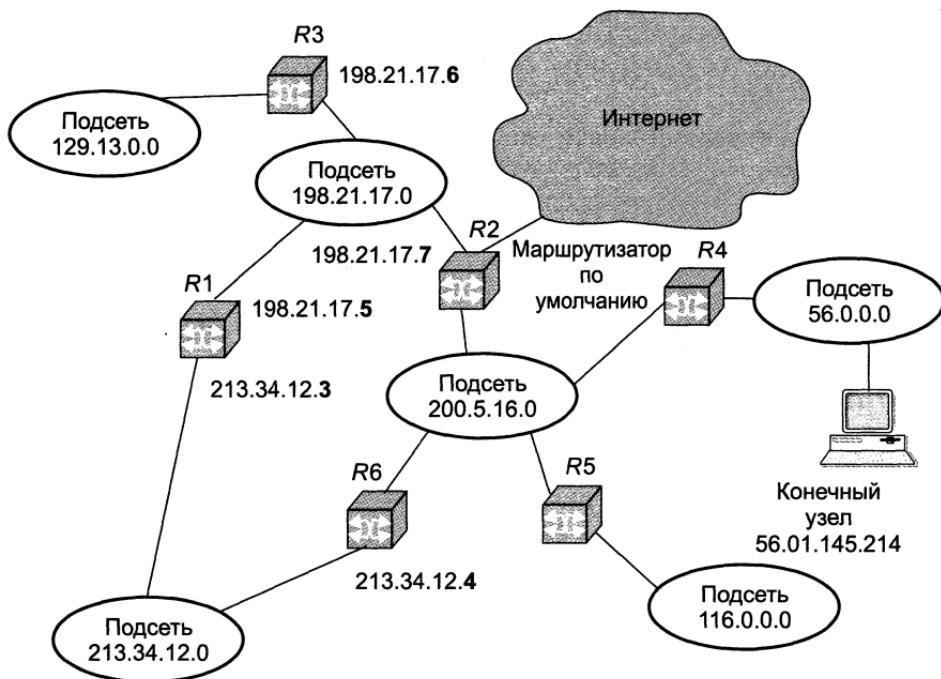


Рис. 3.8. Пример маршрутизируемой сети

В сложных составных сетях почти всегда существуют несколько альтернативных маршрутов для передачи пакетов между двумя конечными узлами. Так, пакет, отправленный из сети 129.13.0.0 в сеть 56.0.0.0, может быть доставлен двумя путями. 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 количество пройденных на маршруте промежуточных маршрутизаторов (*ретрансляционных участков*, или *хопов*). Полученная в результате анализа информация о маршрутах помещается в **таблицу маршрутизации**.

Посмотрим, как могла бы выглядеть таблица маршрутизации, например, в маршрутизаторе R_1 (табл. 3.3). Составляя эту таблицу, мы предполагаем, что в сети используется адресация на основе классов.

ПРИМЕЧАНИЕ

Таблица 3.3 значительно упрощена по сравнению с реальными таблицами, например, здесь отсутствуют столбцы с масками, признаками состояния маршрута, временем, в течение которого действительны записи данной таблицы (их применение будет рассмотрено позже). Вместо номера сети назначения может быть указан полный сетевой адрес отдельного узла назначения.

Таблица 3.3. Упрощенная таблица маршрутизации маршрутизатора R1

Адрес назначения	Адрес следующего маршрутизатора	Адрес выходного интерфейса	Расстояние до сети назначения
56.0.0.0	213.34.12.4	213.34.12.3	2
116.0.0.0	213.34.12.4	213.34.12.3	2
129.13.0.0	198.21.17.6	198.21.17.5	1
198.21.17.0	198.21.17.5	198.21.17.5	0 – непосредственно подключена
213.34.12.0	213.34.12.3	213.34.12.3	0 – непосредственно подключена
200.5.16.0	213.34.12.4	213.34.12.3	1
56.01.145.214	198.21.17.7	198.21.17.5	2
Маршрут по умолчанию	198.21.17.7	198.21.17.5	—

Первый столбец таблицы содержит **адреса назначения** пакетов. 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. 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. Если в таблице имеются записи о маршрутах как к сети в целом, так и к еециальному узлу, то при поступлении пакета, адресованного данному узлу, маршрутизатор отдаст предпочтение специальному маршруту. Так, в столбце адреса назначения табл. 3.3, наряду с адресом сети 56.0.0.0 (класс A), имеется отдельная запись об одном принадлежащем этой сети узле (56.01.145.214), для которого определен специфический маршрут.

В каждой строке таблицы следом за адресом назначения указывается **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 (198.21.17.5 или 213.34.12.3) 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 расстояния до сети назначения. При этом расстояние измеряется в любой метрике, используемой в соответствии с заданным в сетевом пакете критерием, в частности, это может быть время прохождения пакета по линиям связи, различные характеристики надежности линий связи на данном маршруте, пропускная способность или другая величина, отражающая качество данного маршрута по отношению к заданному критерию. В табл. 3.3 расстояние между сетями измеряется хопами, то есть количеством транзитных маршрутизаторов. Расстояние для сетей, непосредственно подключенных к портам маршрутизатора (в нашем примере это две сети, 198.21.17.0 и 213.34.12.0, класса C), здесь принимается равным 0, однако в некоторых реализациях отсчет расстояний начинается с 1.

Поскольку в общем случае 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. Но 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). В этом приеме используются особенности топологии сети. Например, в таблицах маршрутизаторов, находящихся на периферии составной сети, достаточно записать номера только тех сетей, которые непосредственно подсоединенены к ним или расположены поблизости, на тупиковых маршрутах. Обо всех же 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 в таблице маршрутизатора R1 установлены специфические маршруты только для пакетов, следующих в сети, адреса которых явно указаны на рисунке. Для всех остальных пакетов, направляющихся по самым разным адресам Интернета, в таблице маршрутизатора R1 указан один и тот же адрес входного интерфейса следующего маршрутизатора R2 (198.21.17.7), который в данном случае и является маршрутизатором по умолчанию.

Задачу маршрутизации решают не только маршрутизаторы, но и **конечные узлы** — компьютеры. Когда на конечном узле данные, подлежащие передаче, поступают в распоряжение протокола IP, он, прежде всего, определяет,

направляется ли пакет в другую сеть или адресован какому-нибудь узлу той же сети, к которой относится данный конечный узел. Если номера сети назначения и сети отправителя совпадают, это означает, что пакет маршрутизировать не требуется.

В противном случае маршрутизация нужна, и для решения этой задачи привлекается таблица маршрутизации конечного узла-отправителя. 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, основываясь только на сведениях об адресе маршрутизатора по умолчанию. Если в сети имеется только один маршрутизатор, этот вариант — единственно возможный для всех конечных узлов. Но даже при наличии в сети нескольких маршрутизаторов, когда перед конечным узлом стоит проблема их выбора, часто в компьютерах для повышения производительности прибегают к заданию маршрута по умолчанию.

Далее перечислены основные источники записей таблицы маршрутизации.

- **Программное обеспечение стека TCP/IP** 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.
- **Администратор** заносит записи в таблицу маршрутизации вручную, используя команды конфигурирования маршрутизатора. Чаще всего вручную заносятся записи о маршруте по умолчанию и о специфическом для узла маршруте. Заданные вручную записи всегда являются *статическими*, то есть постоянно сохраняются в таблице.
- **Протоколы маршрутизации**, такие как RIP или OSPF, формируют записи о возможных маршрутах автоматически. Эти записи являются *динамическими*, то есть имеют ограниченный срок жизни.

Маршрутизация без масок

На рис. 3.9 приведен алгоритм просмотра таблицы маршрутизации протоколом IP, установленным на маршрутизаторе. При его описании мы будем ссылаться на табл. 3.3 и рис. 3.8.

1. Пусть на один из интерфейсов маршрутизатора поступает пакет с адресом назначения 56.01.145.214. Протокол IP извлекает этот адрес из пакета.

2. Выполняется *первая фаза* просмотра таблицы – *поиск специфического маршрута* к адресу назначения. IP-адрес (целиком, и номер сети, и номер узла) последовательно строка за строкой сравнивается с содержимым поля адреса назначения таблицы маршрутизации. Если произошло совпадение (как в табл. 3.3), то из соответствующей строки извлекаются адрес следующего маршрутизатора (198.21.17.7) и идентификатор выходного интерфейса (198.21.17.5). На этом просмотр таблицы заканчивается.

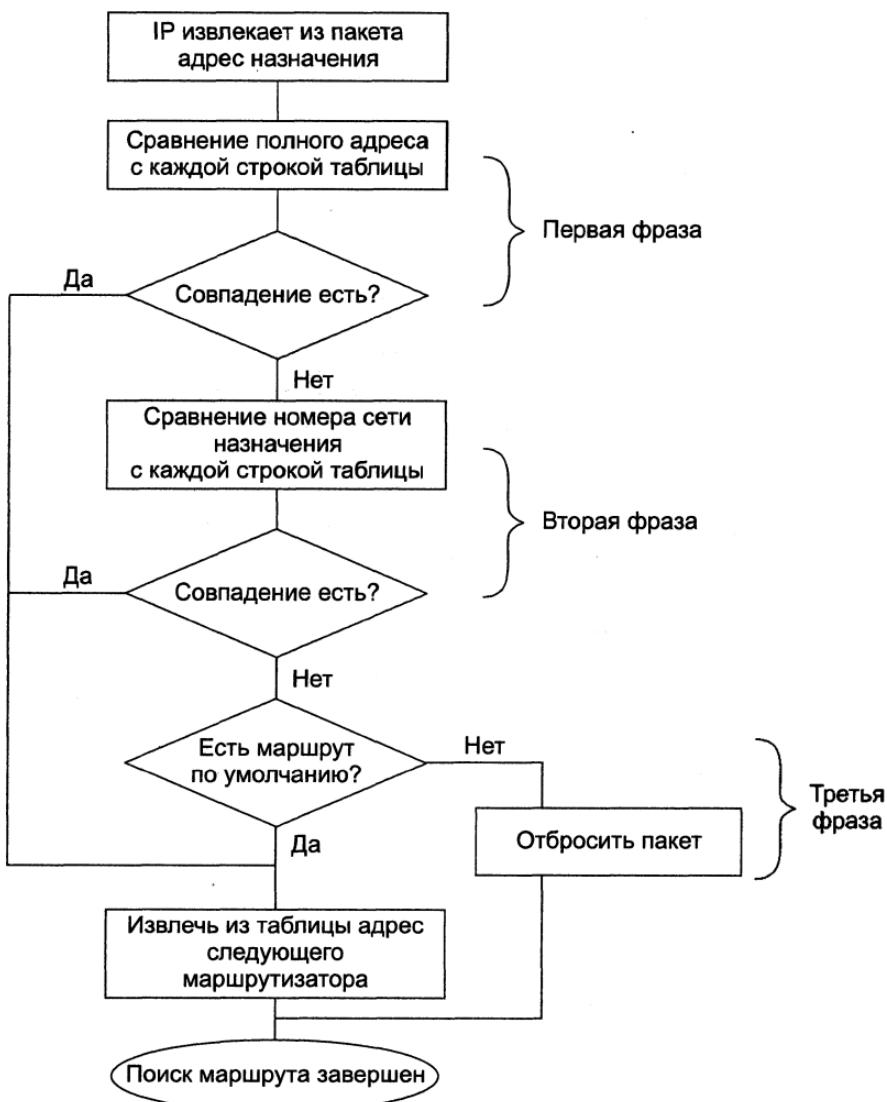


Рис. 3.9. Алгоритм просмотра таблицы маршрутизации

3. Предположим теперь, что в таблице не нашлось строки, в которой адрес назначения полностью совпал бы с адресом назначения, извлеченным из пакета. Такая ситуация возникла бы, например, если бы в пакете был указан адрес 56.01.15.24. В таком случае совпадения не произошло бы, и протокол IP перешел ко *второй фазе* просмотра — *поиску маршрута к сети назначения*. Для этого из IP-адреса выделяется номер сети. В нашем примере из адреса 56.01.15.24 класса A выделяется номер сети 56.0.0.0, и таблица снова просматривается на предмет совпадения данного номера сети с адресом назначения в какой-либо строке. При совпадении (а в нашем примере оно произошло в первой строке) из таблицы извлекаются адрес следующего маршрутизатора (213.34.12.4) и идентификатор выходного интерфейса (213.34.12.3). Просмотр таблицы на этом завершается.
4. Наконец, предположим, что адрес назначения в пакете был таков, что совпадения не произошло ни в первой, ни во второй фазах просмотра. В таком случае выполняется *третья фаза* — протокол IP либо выбирает маршрут по умолчанию (и пакет направляется по адресу 198.21.17.7), либо, если маршрут по умолчанию отсутствует, отбрасывает пакет¹. 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.

Пример взаимодействия протоколов IP, ARP, Ethernet и DNS

Рассмотрим процесс продвижения пакета в составной сети на примере IP-сети, показанной на рис. 3.10. При этом будем считать, что все узлы сети, рассматриваемой в примере, имеют адреса, основанные на классах. Особое внимание будет уделено взаимодействию протокола IP с протоколами разрешения адресов ARP и DNS.

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
```

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 даже маршрут по умолчанию.

Эта команда включает три последовательные операции: передача от клиента DNS-запроса для определения IP-адреса узла назначения, передача от сервера DNS-ответа о найденном IP-адресе FTP-сервера, передача пакета от FTP-клиента к FTP-серверу. Давайте последовательно, по шагам, рассмотрим, как взаимодействуют между собой протоколы DNS, IP, ARP и Ethernet и что происходит с кадрами и пакетами при решении этих задач.

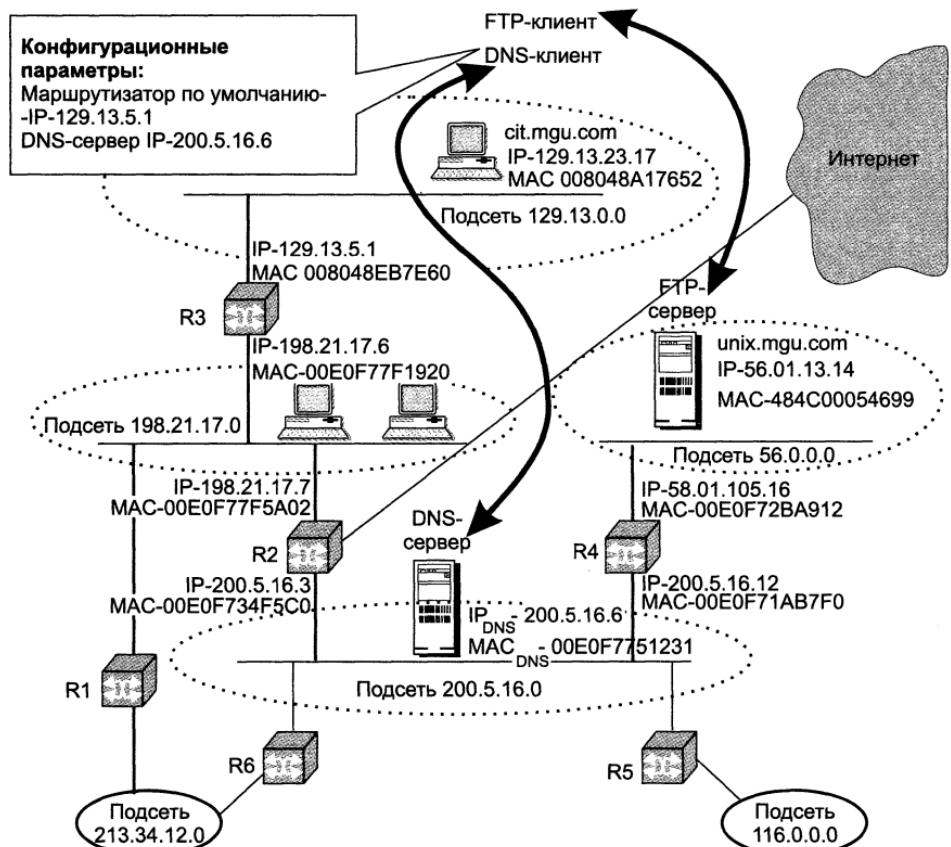


Рис. 3.10. Пример IP-маршрутизации

1. **Формирование IP-пакета с инкапсулированным в него DNS-запросом.** Программный модуль FTP-клиента, получив команду `> ftp unix.mgu.com`, передает запрос к работающей на этом же компьютере клиентской части протокола DNS, которая, в свою очередь, формирует к DNS-серверу запрос, интерпретируемый примерно так: «Какой IP-адрес соответствует символьному 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. Этот адрес

известен программному обеспечению клиентского компьютера, так как он входит в число его конфигурационных параметров. Сформированный IP-пакет будет перемещаться по сети в неизменном виде (как показано на рис. 3.11), пока не дойдет до адресата – DNS-сервера.

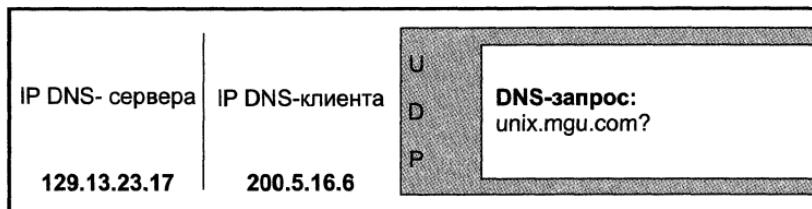


Рис. 3.11. IP-пакет с инкапсулированным в него DNS-запросом

2. *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. Для этого 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 для кадра 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 194.87.23.1?» – 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 *sit.mgu.com* отправляет маршрутизатору R3 пакет, упакованный в кадр Ethernet (рис. 3.12).
3. *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

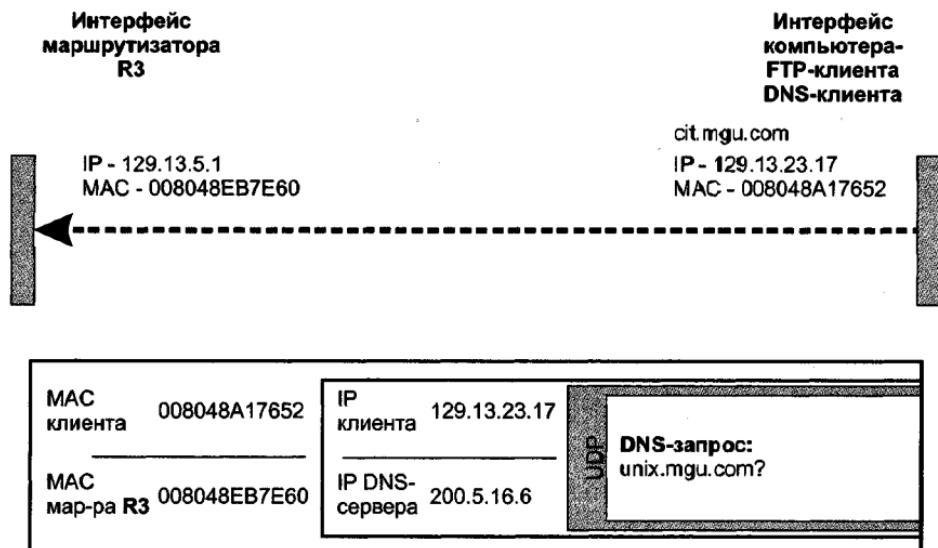


Рис. 3.12. Кадр Ethernet с инкапсулированным IP-пакетом, отправленный с клиентского компьютера

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 поступают на интерфейс следующего маршрутизатора R2, имеющего IP-адрес 198.21.17.7. Однако знания IP-адреса недостаточно, чтобы передать пакет по сети Ethernet. Необходимо определить MAC-адрес маршрутизатора R3. Как известно, такой работой занимается протокол ARP. Пусть на этот раз в ARP-таблице нет записи об адресе маршрутизатора R3. Тогда в сеть отправляется широковещательный ARP-запрос, который поступает на все интерфейсы сети 198.21.17.0. Ответ приходит только от интерфейса маршрутизатора R3: «Я имею IP-адрес 198.21.17.7 и мой MAC-адрес 00E0F77F5A02» (рис. 3.13).

Теперь, зная MAC-адрес маршрутизатора R2 (00E0F77F5A02), маршрутизатор R3 отсылает ему IP-пакет с DNS-запросом (рис. 3.14).

4. *Маршрутизатор R2 доставляет пакет DNS-серверу.* Модуль IP на маршрутизаторе R2 действует в соответствии с уже не раз описанной нами процедурой: отбросив заголовок кадра Ethernet, он извлекает из пакета IP-адрес назначения и просматривает свою таблицу маршрутизации. Там он 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 из ARP-таблицы, либо по запросу) требуемый MAC-адрес 00E0F7751231 DNS-сервера. Получив ответ о MAC-адресе, маршрутизатор R2 отправляет в сеть назначения кадр Ethernet с DNS-запросом (рис. 3.15).

**Интерфейс
маршрутизатора
R2**

IP - 198.21.17.7
MAC - 00E0F77F5A02

**Интерфейс
маршрутизатора
R3**

IP - 129.13.5.1
MAC - 008048EB7E60

Кадры Ethernet

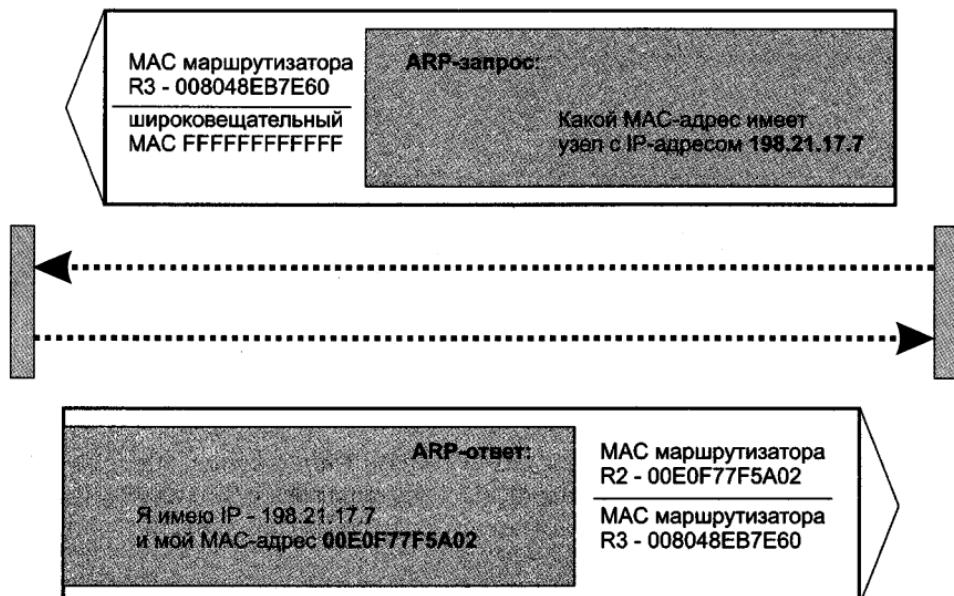


Рис. 3.13. Кадры Ethernet с инкапсулированными ARP-запросом и ARP-ответом

**Интерфейс
маршрутизатора
R2**

IP - 198.21.17.7
MAC - 00E0F77F5A02

**Интерфейс
маршрутизатора
R3**

IP - 129.13.5.1
MAC - 008048EB7E60

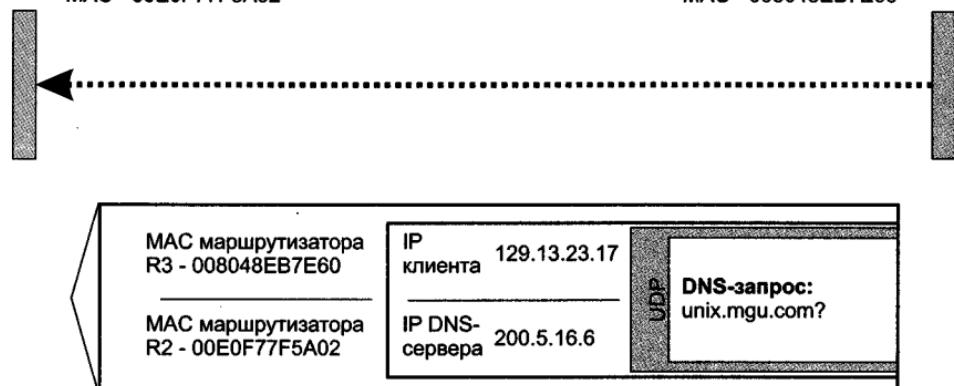


Рис. 3.14. Кадр Ethernet с DNS-запросом, отправленный с маршрутизатора R3 маршрутизатору R2

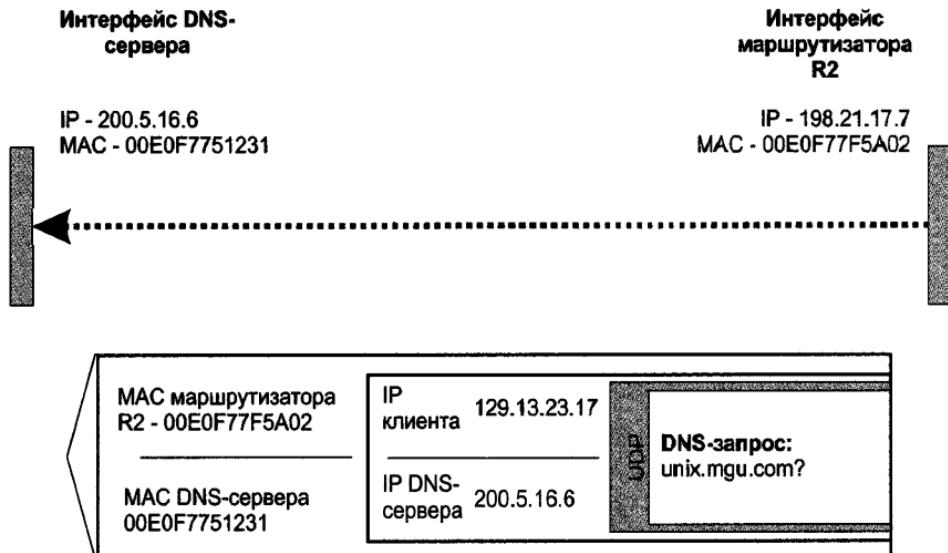


Рис. 3.15. Кадр Ethernet с DNS-запросом, отправленный с маршрутизатора R2

5. Сетевой адаптер DNS-сервера захватывает кадр Ethernet, обнаруживает совпадение MAC-адреса назначения, содержащегося в заголовке, со своим собственным адресом и направляет его модулю IP. После анализа полей заголовка IP из пакета извлекаются данные вышеперечисленных протоколов. DNS-запрос передается программному модулю DNS-сервера. DNS-сервер просматривает свои таблицы, возможно, обращается к другим DNS-серверам, и в результате формирует ответ, смысл которого состоит в следующем: «Символьному имени unix.mgu.com соответствует IP-адрес 56.01.13.14».

ПРИМЕЧАНИЕ

Заметим, что во время всего путешествия пакета по составной сети от клиентского компьютера до DNS-сервера адреса получателя и отправителя в полях заголовка IP-пакета не изменились. Зато в заголовке каждого нового кадра, который переносил пакет от одного маршрутизатора к другому, MAC-адреса отправителя и получателя изменялись на каждом отрезке пути.

Процесс доставки DNS-ответа совершенно аналогичен процессу передачи DNS-запроса, который мы только что так подробно описали. Работая в тесной кооперации, протоколы IP, ARP и Ethernet передают FTP-клиенту DNS-ответ через всю составную сеть (рис. 3.16).

FTP-клиент, получив IP-адрес FTP-сервера, посыпает ему свое сообщение, используя те же механизмы доставки данных через составную сеть, которые были описаны. Однако для читателя будет весьма полезно мысленно воспроизвести этот процесс, обращая особое внимание на значения адресных полей заголовков кадров и заголовка вложенного IP-пакета.

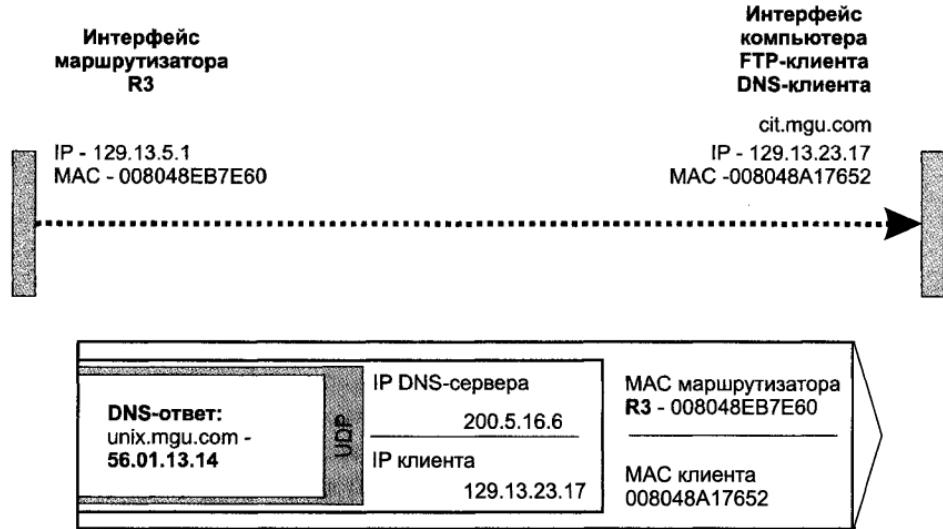


Рис. 3.16. Кадр Ethernet с DNS-ответом, отправленный с маршрутизатора R3 компьютеру-клиенту

Структуризация сетей на основе масок

Алгоритм маршрутизации усложняется, когда в систему адресации узлов вносятся дополнительные элементы — маски. Использование масок, как мы уже изучили, делает более гибкой процедуру распределения адресов, поэтому клиенты часто получают от поставщиков услуг пулы адресов в виде *IP-адрес/маска*. Кроме того, и сами администраторы часто применяют маски для структуризации своих сетей. Разделения одной сети на подсети позволяет сделать более простой задачу администрирования. Во-первых, в таком случае легче распределить функции управления сетью между несколькими администраторами. Во-вторых, многие задачи по контролю сетевого трафика могут быть возложены на маршрутизаторы, которые, будучи пограничными устройствами, способны фильтровать трафик автоматически, руководствуясь сложными формальными правилами¹.

Допустим, администратор получил в свое распоряжение сеть 129.44.0.0 класса В (2^{16} узлов) или, что в данном случае одно и то же, пул адресов 129.44.0.0/16. Такое адресное пространство позволяет администратору организовать одну большую неструктурированную сеть, однако производственная необходимость требует, чтобы сеть была разделена на подсети разного размера, причем трафик в каждой подсети надежно локализован. Это позволяет легче администрировать и диагностировать сеть, а также проводить в каждой из подсетей особую политику безопасности. (Заметим, что разделение

¹ Мы рассмотрим фильтрацию, осуществляемую маршрутизаторами, немного позже в этой главе.

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. 3.17 приведен пример решения поставленной задачи. Половина (2^{16}) из имеющихся 2^{16} адресов отведена для создания сети 1, имеющей префикс 129.44.0.0/17. Следующая порция адресов, составляющая четверть всего адресного пространства (2^{14}), назначена для сети 2 (129.44.128.0/18). Далее в пространстве адресов был «вырезан» небольшой фрагмент для создания сети 3, предназначенный 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, в то время как оставшиеся два (10 и 01) позволяют адресовать порты маршрутизаторов. Поле номера узла в таком случае имеет два двоичных разряда, и пул адресов определяется как 129.44.192.0/30.

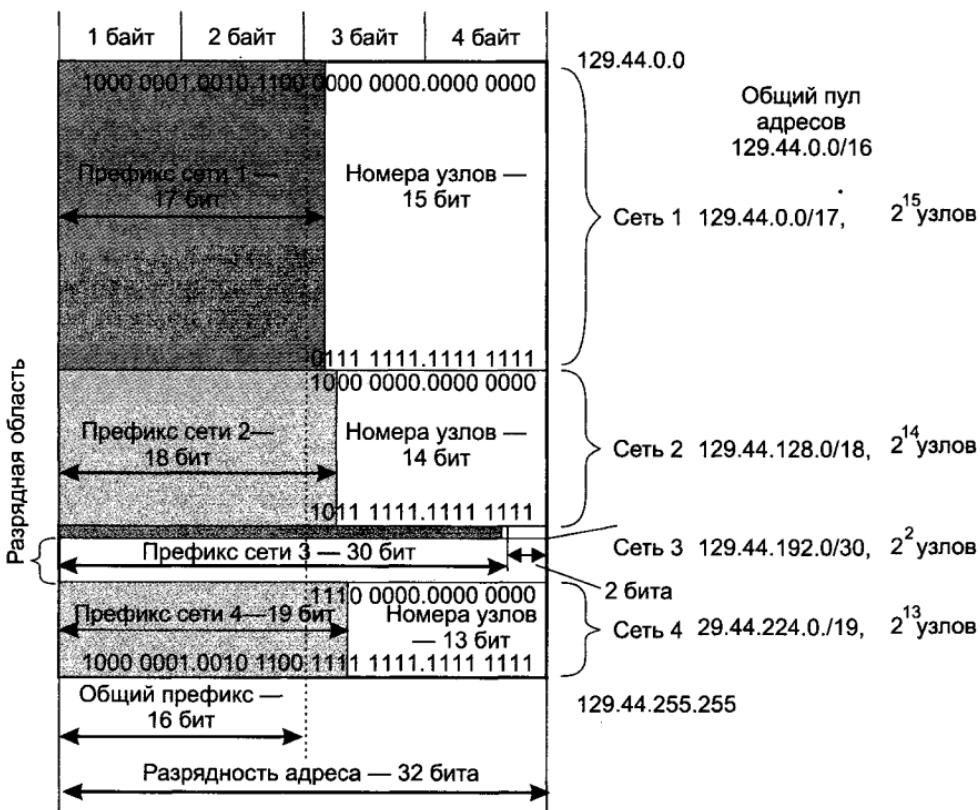


Рис. 3.17. Разделение адресного пространства сети 129.44.0.0 класса В на сети разного размера с помощью масок переменной длины

Из оставшегося пула ($2^{14} - 4$) адресов администратор организовал еще одну большую сеть 3 (129.44.192.0/19) с числом узлов 2^{13} . Но даже после этого имеющийся «остаток» оказался так велик, что из него можно было бы образовать еще 31 сеть, каждая из которых равна размеру стандартной сети класса С. Однако эту область адресов администратор решил оставить в резерве. Таким образом, мы видим, что маски переменного размера дают администратору огромные возможности в плане гибкого использования имеющегося у него адресного пространства.

На рис. 3.18 показан пример составной сети, построенной путем связывания маршрутизаторами сетей 1, 2, 3 и 4. Несмотря на относительно сложную структуру, извне эта составная сеть по-прежнему выглядит как единая сеть класса В. Весь трафик, выходящий из этой составной сети «наружу» (за пределы маршрутизатора R1) и входящий в нее может маршрутизоваться на основе классов. Внутри сети трафик обрабатывается четырьмя маршрутизаторами с учетом назначенных администратором масок.

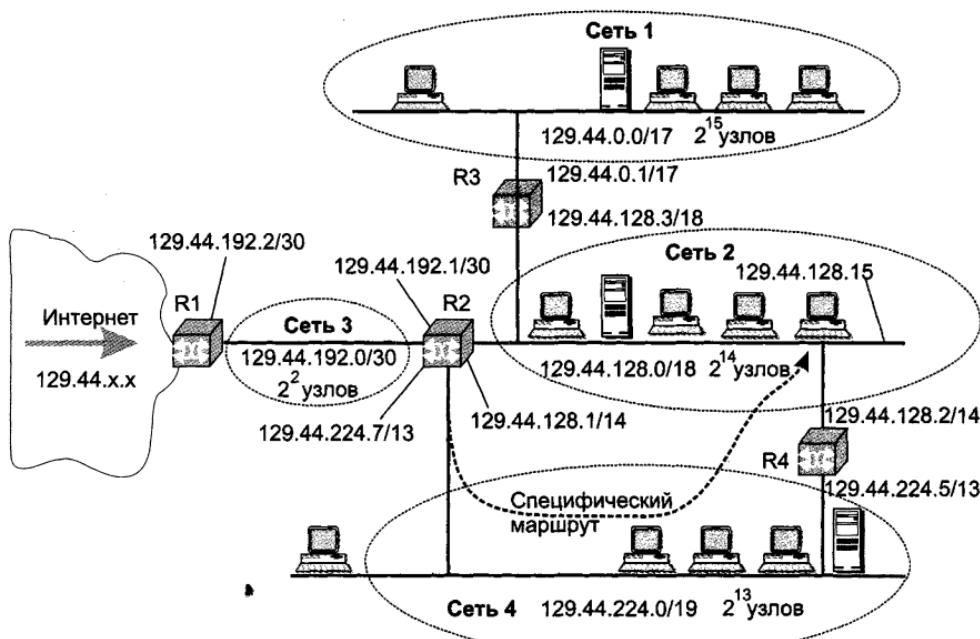


Рис. 3.18. Структуризация сети масками переменной длины

Маршрутизация с масками

При использовании масок, так же как и при адресации, основанной на классах, главным критерием, на основании которого пакет направляется по тому или иному маршруту, является номер сети. Имея дело с адресами стандартных классов, маршрутизатор легко находит границу между номером сети

и номером узла. Однако адресация с использованием масок усложняет эту процедуру. Важно подчеркнуть, что в IP-пакетах, путешествующих по сети, содержатся только IP-адреса обычного вида, то есть значения масок в них не указаны. Так что маршрутизатор, извлекая адрес из IP-пакета, не может определить, какая часть 32-разрядного числа, помещенного в поле адреса назначения, является номером сети, а какая — номером узла. Ответ на этот вопрос маршрутизатор получает из таблицы маршрутизации, в которую добавляется столбец маски.

В табл. 3.4 показана таблица маршрутизатора *R2* из нашего последнего примера (см. рис. 3.18). Вспомогательный столбец «Примечание» добавлен в эту таблицу в учебных целях.

Таблица 3.4. Таблица маршрутизатора *R2* в сети с масками переменной длины

Примечание	Адрес назначения	Маска	Адрес следующего маршрутизатора	Адрес выходного интерфейса	Расстояние
Сеть 1, префикс /17	129.44.0.0	255.255.128.0	129.44.128.3	129.44.128.1	1
Сеть 2, префикс /18	129.44.128.0	255.255.192.0	—	129.44.128.1	Подключена
Сеть 3, префикс /30	129.44.192.0	255.255.255.252	—	129.44.192.1	Подключена
Сеть 4, префикс /19	129.44.224.0	255.255.224.0		129.44.224.7	Подключена
Маршрут по умолчанию	0.0.0.0	0.0.0.0	129.44.192.2	129.44.192.1	—
Специфический маршрут	129.44.128.15	255.255.255.255	129.44.224.7	129.44.224.5	

Первые четыре записи в таблице соответствуют маршрутам к внутренним подсетям. 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. По каким-то причинам администратор сети решил назначить для трафика, направляющегося к этому узлу, особый, «обходной» маршрут. В отличие от всех других узлов сети 129.44.128.0/18, к которым пакеты поступают с интерфейса 129.44.128.1 маршрутизатора *R2*, к данному узлу они будут приходить через маршрутизатор *R4*. В тех строках таблицы, в которых в качестве адреса назначения указан полный IP-адрес узла, маска всегда имеет значение 255.255.255.255.

Алгоритм просмотра таблиц маршрутизации, содержащих маски, имеет много общего с описанным ранее алгоритмом просмотра таблиц, не содержащих маски. Однако имеются и существенные различия.

1. 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 процедуре просмотра таблицы маршрутизации, также состоящей из двух фаз, как и процедура просмотра таблицы, в которой столбец маски отсутствует.
2. *Первая фаза* состоит в поиске специфического маршрута для адреса IP_D . С этой целью из каждой строки таблицы, в которой маска имеет значение 255.255.255.255, извлекается адрес назначения и сравнивается с адресом IP_D . Если в какой-либо строке совпадение произошло, то адрес следующего маршрутизатора для обрабатываемого пакета берется из данной строки и на этом поиск маршрута для данного адреса заканчивается.
3. *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 IP заново просматривает таблицу маршрутизации, причем с каждой записью (включая строку о маршруте по умолчанию) производятся следующие действия:
 - а) маска из очередной строки таблицы (обозначим ее M) «накладывается» на IP_D , то есть выполняется операция $IP_D \text{ AND } M$;
 - б) полученное в результате число сравнивается со значением, которое помещено в поле адреса назначения той же записи таблицы маршрутизации;
 - в) если происходит совпадение, протокол IP соответствующим образом отмечает эту строку;
 - г) если просмотрены не все строки, то протокол IP переходит к следующей строке, если все, то просмотр записей заканчивается и происходит переход к следующему шагу.
4. В заключение маршрутизатор выполняет одно из трех действий:
 - а) если не произошло ни одного совпадения и маршрут по умолчанию отсутствует, то пакет отбрасывается;
 - б) если произошло одно совпадение, то пакет отправляется по маршруту, указанному в строке с совпавшим адресом;
 - в) 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).

Проиллюстрируем, как маршрутизатор R2 реализует описанный алгоритм при работе со своей таблицей маршрутизации. Пусть поступивший на R2 пакет имеет адрес назначения 129.44.230.15. Модуль IP, установленный на этом

маршрутизаторе, прежде всего сравнив этот адрес с адресом 129.44.128.15, для которого определен специфический маршрут. Совпадения нет, поэтому модуль IP начинает последовательно обрабатывать все строки таблицы, накладывая маски, сравнивая результаты и отмечая совпадения, до тех пор, пока не просмотрит все записи таблицы. Мы свели результаты этих вычислений в табл. 3.5.

Таблица 3.5. Результаты вычислений маршрутизатора R2 во время просмотра таблицы

Адрес назначения из таблицы маршрутизации	Наложение маски на адрес назначения из пакета	Номер сети, вычисленный наложением маски на адрес из пакета	Результат сравнения адреса из таблицы с вычисленным номером
129.44.0.0	255.255.128.0 AND 129.44.230.15	129.44.128.0	Нет совпадения
129.44.128.0	255.255.192.0 AND 129.44.230.15	129.44.192.0	Нет совпадения
129.44.192.0	255.255.255.252 AND 129.44.230.15	129.44.230.12	Нет совпадения
129.44.224.0	255.255.224.0 AND 129.44.230.15	129.44.224.0	Есть совпадение

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 соответствует адресу, указанному в записи. Поскольку маска 0.0.0.0 имеет нулевую длину, то этот маршрут считается самым неспецифическим и используется только при отсутствии совпадений с остальными записями из таблицы маршрутизации.

Таким образом, совпадение имеет место в одной строке, которой соответствует строка таблицы маршрутизатора табл. 3.5, указывающая на то, что данный пакет должен быть передан на выходной интерфейс 129.44.224.7 маршрутизатора R2 и отправлен в непосредственно подключенную к нему сеть 4.

CIDR и маршрутизация

За последние годы в Интернете многое изменилось: резко возросло число узлов и сетей, повысилась интенсивность трафика, изменился характер передаваемых данных. Из-за несовершенства протоколов маршрутизации обмен сообщениями об обновлении таблиц стал приводить к сбоям магистральных маршрутизаторов, происходящим из-за перегрузок при обработке большого объема служебной информации. Так, сегодня таблицы магистральных

маршрутизаторов в Интернете могут содержать до нескольких сотен и даже тысяч маршрутов.

На решение этой проблемы направлена, в частности, уже знакомая нам технология CIDR.

Согласно рекомендациям технологии CIDR, для повышения эффективности маршрутизации поставщикам услуг Интернета назначается *непрерывный* диапазон IP-адресов, в результате все адреса каждого поставщика услуг получают общий префикс. Следовательно, маршрутизация на магистралях Интернета может осуществляться на основе префиксов, а не полных адресов сетей. А это значит, что вместо множества (по числу сетей) записей будет достаточно поместить *одну запись* сразу для всех сетей, имеющих общий префикс. Такое *агрегирование* адресов позволяет уменьшить объем таблиц маршрутизации на маршрутизаторах всех уровней, а следовательно, ускорить работу маршрутизаторов и повысить пропускную способность Интернета.

Ранее мы рассмотрели пример, где администратор использовал маски для деления непрерывного пула адресов, полученного от поставщика услуг, на несколько частей, обеспечивая структуризацию своей сети. Такой вариант применения масок называется *разделением на подсети* (subnetting).

Вместе с тем при разделении сети на подсети с помощью масок проявлялся и обратный эффект – объединение подсетей. Упрощенно говоря, для того чтобы направить весь суммарный трафик, адресованный из внешнего окружения в корпоративную сеть, разделенную на подсети, достаточно, чтобы в таблицах маршрутизации всех внешних маршрутизаторов имелась только 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 *агрегирования нескольких сетей в одну более крупную сеть* (supernetting).

Итак, внедрение технологии CIDR позволяет решить две основные задачи.

- Более экономно расходовать адресное пространство. Благодаря технологии CIDR поставщики услуг получают возможность «нарезать» блоки из выделенного им адресного пространства в точном соответствии с требованиями каждого клиента, при этом у клиента остается пространство для маневра на случай будущего роста.
- Уменьшить число записей в таблицах маршрутизации за счет объединения маршрутов – одна запись в таблице маршрутизации может представлять большое количество сетей.

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, до сих пор распределение адресов носит во многом случайный характер. Кардинальный путь решения проблемы — перенумеровать сети. Однако эта процедура сопряжена с определенными временными 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 больше не фигурировал в магистральных маршрутизаторах. Требование оплаты за каждый адрес узла также может подтолкнуть пользователя решиться на перенумерование с тем, чтобы получить ровно столько адресов, сколько ему нужно.

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 одного и того же стека внутри компьютера. Протоколы, выполняющие фрагментацию в пределах узла, анализируют тип технологии нижнего уровня, определяют ее MTU и делят сообщения на такие части, которые умещаются в кадры канального уровня того же стека протоколов.

В стеке TCP/IP эту задачу решает протокол TCP, который разбивает поток байтов, передаваемый 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 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.

Каждый из фрагментов должен быть снабжен полноценным заголовком IP. Некоторые из полей заголовка (идентификатор, TTL, флаги DF и MF, смещение) непосредственно предназначены для последующей *сборки* фрагментов в исходное сообщение.

- **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** (Time To Live, TTL) 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). 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, пакет уничтожается. При сборке фрагментов хост-получатель использует значение TTL как крайний срок ожидания недостающих фрагментов.
- Поле **смещения фрагмента** предоставляет получателю информацию о положении фрагмента относительно начала поля данных исходного нефрагментированного пакета. Так, например, 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 байт.
- 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 нуль. Флаг **DF** (Do not Fragment – не фрагментировать), установленный в единицу, запрещает маршрутизатору фрагментировать данный пакет. Если помеченный таким образом пакет не может достигнуть получателя без фрагментации, то модуль IP его уничтожает, а узлу-отправителю посыпается диагностическое сообщение.

Рассмотрим механизм фрагментации на примере составной сети, показанной на рис. 3.19.

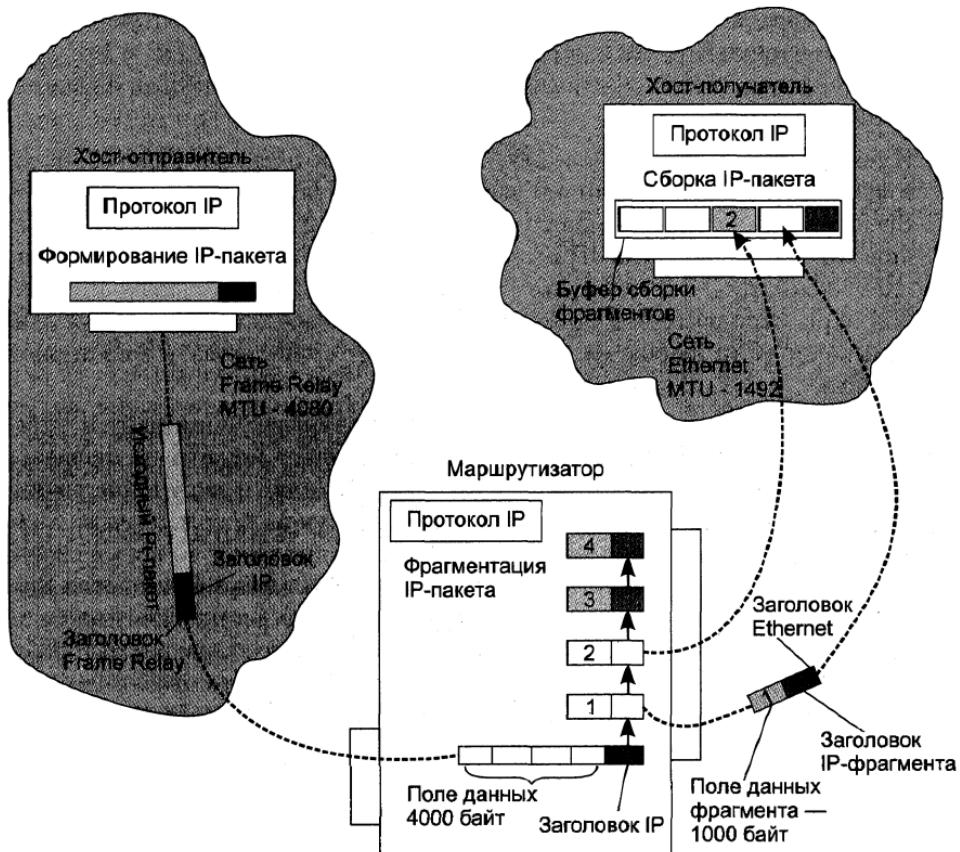


Рис. 3.19. Фрагментация

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, а также на маршрутизаторе, связывающем эти подсети, установлен стек протоколов TCP/IP.

Транспортному уровню *хоста-отправителя* известно значение MTU нижележащей технологии (4080). На основании этого TCP и «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 тех полей заголовка, которые прямо связаны с фрагментацией:

- ❑ пакету присваивается уникальный *идентификатор*, например 12456;
- ❑ поскольку пакет пока еще не был 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 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. Протокол IP выбирает размер поля данных фрагмента равным 1000, так что из одного большого IP-пакета получается 4 маленьких пакета-фрагмента. Для каждого фрагмента и его IP-заголовка в маршрутизаторе создается отдельный буфер (на рисунке фрагменты и соответствующие им буферы пронумерованы от 1 до 4). Протокол IP копирует в эти буфера 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. Процесс фрагментации может изменить значения некоторых полей заголовков IP в пакетах-фрагментах по сравнению с заголовком IP исходного пакета. Так, каждый фрагмент имеет собственное значение контрольной суммы заголовка, смещения фрагмента и общей длины пакета. Во всех пакетах, кроме последнего, флаг MF устанавливается в единицу, а в последнем фрагменте — в нуль. 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.

А 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 IP-адреса отправителя и получателя, а также значения в полях идентификатора (в нашем примере – 12456). Все эти признаки говорят модулю IP, что данные пакеты являются фрагментами одного исходного пакета. Сборка заключается в размещении данных из каждого фрагмента в позицию, определенную *смещением*, указанным в заголовке фрагмента.

Когда первый фрагмент исходного пакета приходит на хост-получатель, этот хост запускает *таймер*, который определяет 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 временных параметров сети, позволяющих оценивать ожидаемое время прибытия фрагментов. Наконец, тайм-аут может быть выбран на базе значений TTL прибывающих фрагментов. Последний подход основан на том, что нет смысла ожидать, пока прибудут другие фрагменты пакета, если время жизни одного из прибывающих фрагментов уже истекло.

ПРИМЕЧАНИЕ

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 кredo «с максимальными усилиями» – стараться по возможности, 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. После того как данные собраны, их можно передать вышележащему протоколу, например TCP.

Протоколы транспортного уровня TCP и UDP

К транспортному уровню стека TCP/IP относятся:

- протокол управления передачей** (Transmission Control Protocol, TCP), RFC 793;
- протокол пользовательских дейтаграмм** (User Datagram Protocol, UDP), RFC 768.

Протоколы TCP и UDP, как и протоколы прикладного уровня, устанавливаются на конечных узлах.

Порты и сокеты

В то время как задачей уровня сетевого взаимодействия, к которому относится протокол IP, является передача данных между сетевыми интерфейсами в составной сети, главная задача протоколов транспортного уровня TCP и UDP заключается в передаче данных между *прикладными процессами*, выполняющимися на компьютерах в сети.

Каждый компьютер может выполнять несколько процессов, более того, даже отдельный прикладной процесс может иметь несколько точек входа, выступающих в качестве адреса 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. Процедура распределения протоколами TCP и UDP поступающих от сетевого уровня пакетов между прикладными процессами называется **демультиплексированием** (рис. 3.20).

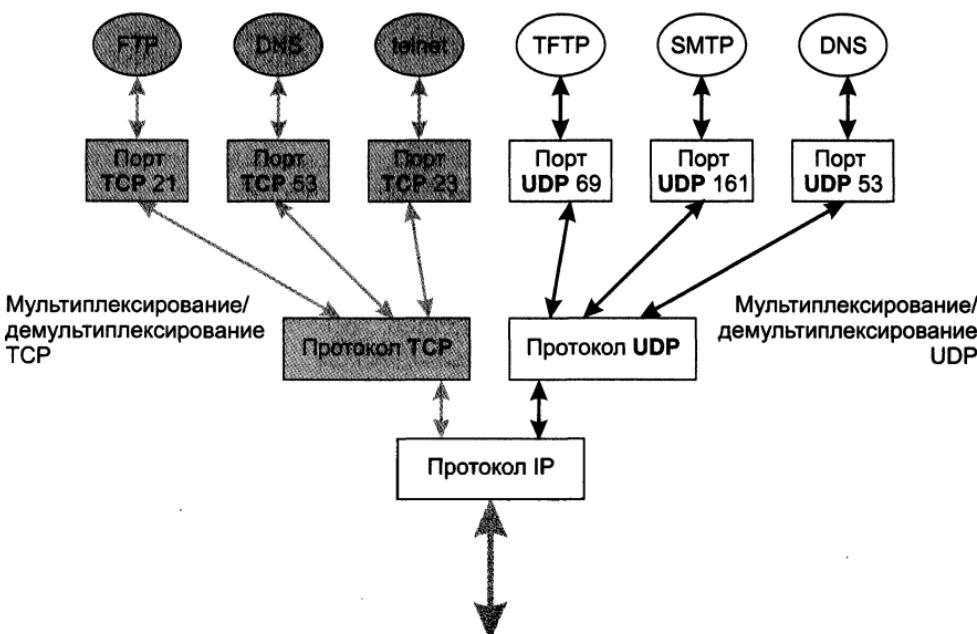


Рис. 3.20. Мультиплексирование и демультиплексирование на транспортном уровне

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 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 операционной системой в ответ на поступление запроса от приложения. На каждом компьютере операционная система ведет список занятых и свободных номеров портов. При поступлении запроса от приложения, выполняемого на данном компьютере, операционная система 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, его номер возвращается в список свободных и может быть назначен другому приложению. Динамические номера являются *уникальными в пределах каждого компьютера*, но при этом обычной ситуацией является совпадение 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 было сказано о портах, в равной степени относится 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, которые передают данные на уровень IP по протоколу UDP, получают номера, называемые **UDP-портами**. Аналогично, приложениям, обращающимся к протоколу TCP, выделяются **TCP-порты**. Совпадение номеров

¹ Порт приложения не надо путать с портами (сетевыми интерфейсами) оборудования.

TCP- и UDP-портов, как правило, ничего не значит. Лишь в некоторых случаях, когда приложение может обращаться по выбору к протоколу TCP или UDP (как, например, DNS-сервер), ему, исходя из удобства запоминания, назначаются совпадающие стандартные номера TCP- и UDP-портов (в данном примере — это хорошо известный номер 53).

Стандартные назначенные номера портов уникально идентифицируют тип приложения (FTP, или HTTP, или DNS и т. д.), однако они 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 — DNS₁, DNS₂ (рис. 3.21). Каждый из этих DNS-серверов имеет хорошо известный UDP-порт 53. Какому из этих серверов нужно было бы направить запрос клиента, если бы в DNS-запросе в качестве идентификатора сервера был указан только номер порта?

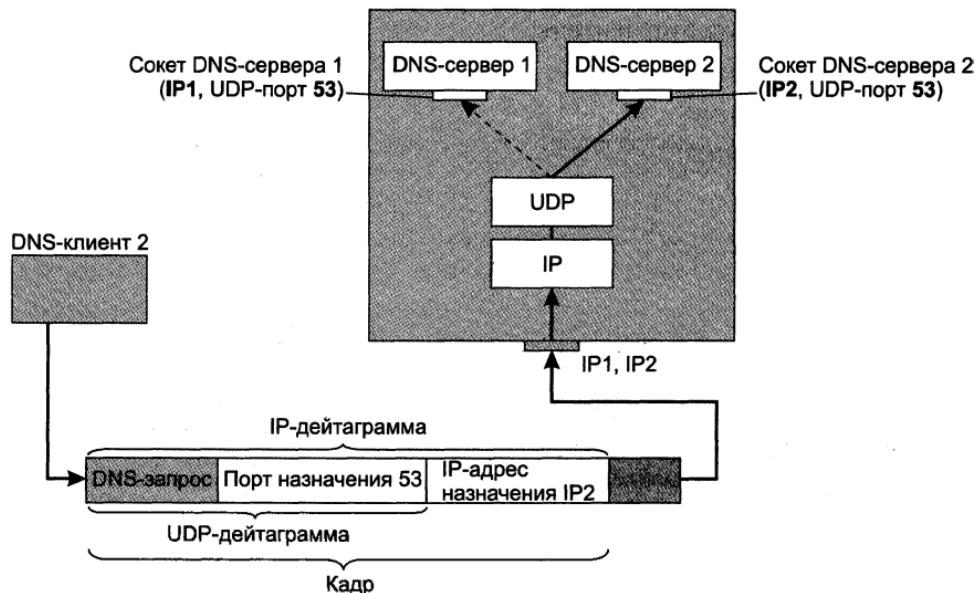


Рис. 3.21. Демультиплексирование протокола UDP на основе сокетов

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 серверу DNS₁, в качестве IP-адреса указывается IP1, а серверу DNS₂ — 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, будет направлен однозначно определенному адресату — DNS₂.

Прикладной процесс однозначно определяется в пределах сети и в пределах отдельного компьютера парой (IP-адрес, номер порта), называемой **сокетом** (socket). Сокет, определенный как IP-адрес, номер UDP-порта, называется **UDP-сокетом**, а как IP-адрес, номер TCP-порта — **TCP-сокетом**.

Протокол UDP

Протокол UDP, подобно IP, является дейтаграммным протоколом, реализующим так называемый ненадежный сервис по возможности, который не гарантирует доставку сообщений адресату.

Заголовок UDP, состоит из четырех 2-байтных полей:

- номер UDP-порта отправителя;
- номер UDP-порта получателя;
- контрольная сумма;
- длина дейтаграммы.

Судя по простоте заголовка, протокол UDP не сложен. Действительно, его функции сводятся к простой передаче данных между прикладным и сетевым уровнями, а также к примитивному контролю искажений в передаваемых данных.

При работе на хосте-отправителе данные от приложений поступают протоколу UDP через порт в виде пользовательских сообщений. 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.

Работая на хосте-получателе, протокол UDP принимает от протокола IP извлеченные из пакетов UDP-дейтаграммы. Извлеченные из IP-заголовка IP-адрес назначения и из UDP-заголовка номер порта образуют сокет приложения. 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.

При контроле искажений протокол UDP только диагностирует, но не исправляет ошибку. Если контрольная сумма показывает, что в поле данных UDP-дейтаграммы произошла ошибка, протокол UDP просто отбрасывает поврежденную дейтаграмму.

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. 3.22). Поступающие данные буферизуются средствами TCP. Для передачи на сетевой уровень из буфера «вырезается» некоторая непрерывная часть данных, которая называется **сегментом**¹ и снабжается заголовком.

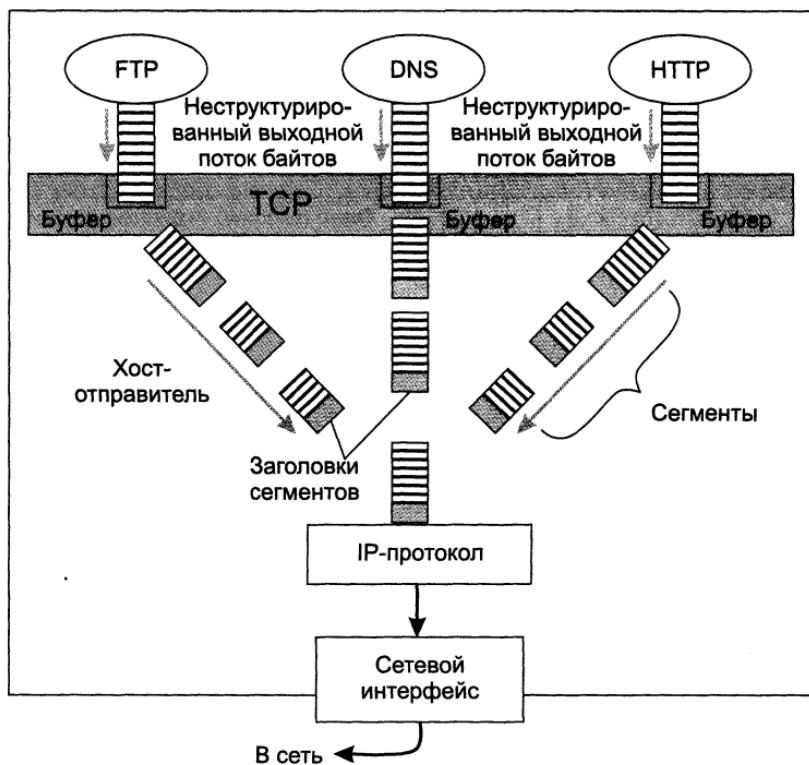


Рис. 3.22. Формирование TCP-сегментов из потока байтов

¹ Заметим, что сегментом называют как единицу передаваемых данных в целом (поле данных и заголовок протокола TCP), так и отдельно поле данных.

ПРИМЕЧАНИЕ

В отличие от протокола UDP, который создает свои 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. 3.23). Мы поместили на рисунок краткое описание большинства полей. Мы рассмотрим их более подробно, когда будем изучать функции протокола TCP.

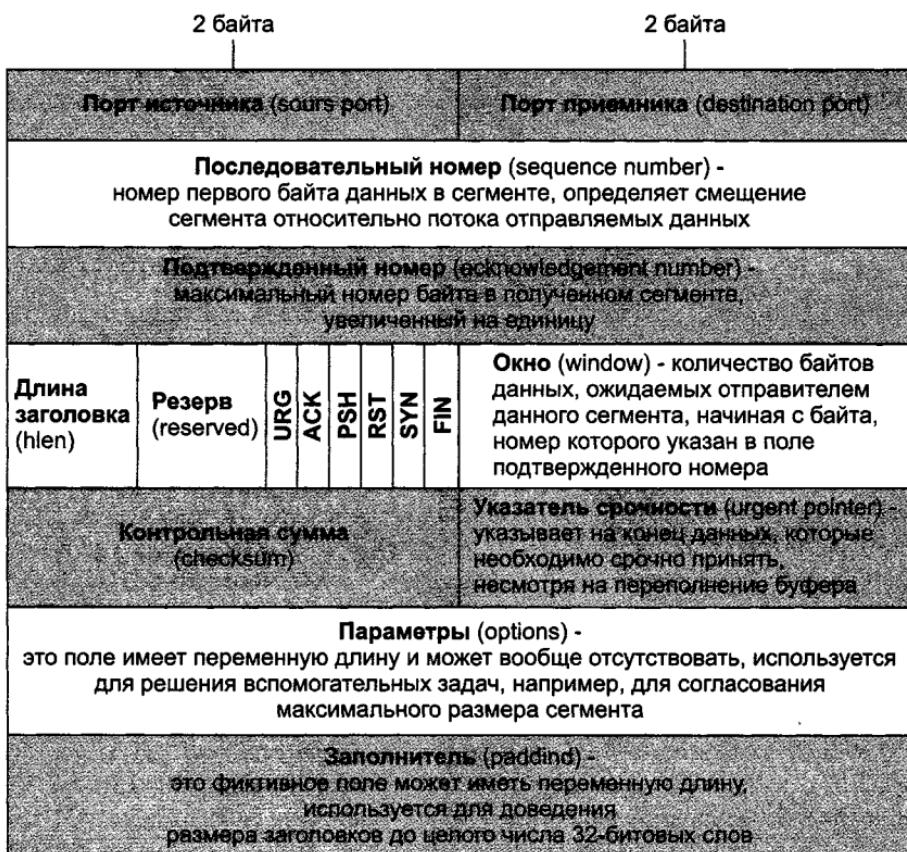


Рис. 3.23. Формат заголовка TCP-сегмента

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 восстановление соединения;
- **SYN** – сообщение, используемое для синхронизации счетчиков переданных данных при установлении соединения;
- **FIN** – признак достижения передающей стороной последнего байта в потоке передаваемых данных.

Логические соединения — основа надежности TCP

В главе 1 мы уже говорили о методе передачи данных с установлением логического соединения. Как мы знаем, 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. В этом разделе мы выясним, в каком виде этот механизм используется в протоколе TCP.

Протокол TCP устанавливает логические соединения между *прикладными процессами*, причем только между *двумя* процессами. TCP-соединение является *дуплексным*, то есть каждый из участников этого соединения может одновременно получать и отправлять данные.

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 посыпает сегмент-запрос на установление соединения протоколу TCP, работающему на стороне сервера (рис. 3.24, 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 операционной системе 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 посыпает клиенту сегмент с флагами ACK и SYN.

В ответ клиент посыпает сегмент с флагом ACK и переходит в состояние установленного логического соединения (состояние ESTABLISHED). Когда сервер получает флаг ACK, он 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. 3.24, б (здесь инициатором является клиент). Соединение считается закрытым по прошествии некоторого времени, в течение которого сторона инициатора убеждается, что его завершающий сигнал ACK дошел нормально и не вызвал никаких «аварийных» сообщений со стороны сервера.

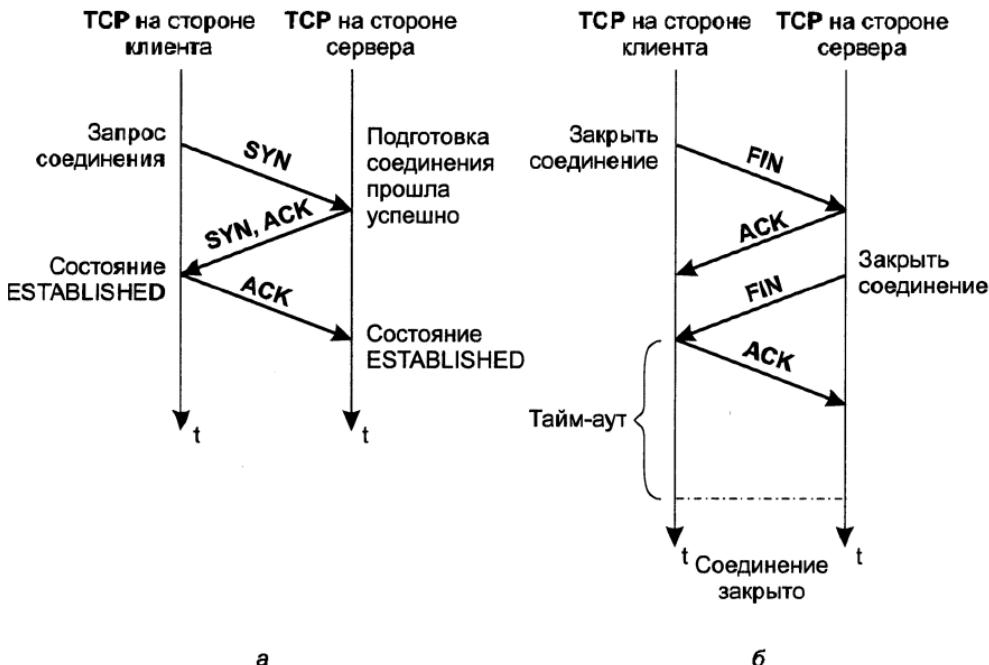


Рис. 3.24. Процедура установления и разрыва логического соединения при нормальном течении процесса

ПРИМЕЧАНИЕ

Мы описали здесь процедуры установления и закрытия соединения очень схематично. Реальные протокольные модули работают в соответствии с гораздо более сложными алгоритмами, учитывающими всевозможные «непштатные» 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. Однако мы скоро вернемся к этим важным деталям работы протокола TCP.

Логическое TCP-соединение однозначно идентифицируется парой сокетов, определенных для этого соединения двумя взаимодействующими процессами.

Сокет одновременно может участвовать в нескольких соединениях. Так, на рис. 3.25 показаны три компьютера с адресами IP1, IP2, IP3. На каждом компьютере выполняется по одному приложению — APPL1, APPL2 и APPL3, сокеты которых, соответственно, (IP1, n1), (IP2, n2), (IP3, n3), а номера TCP-портов приложений — n1, n2, n3.

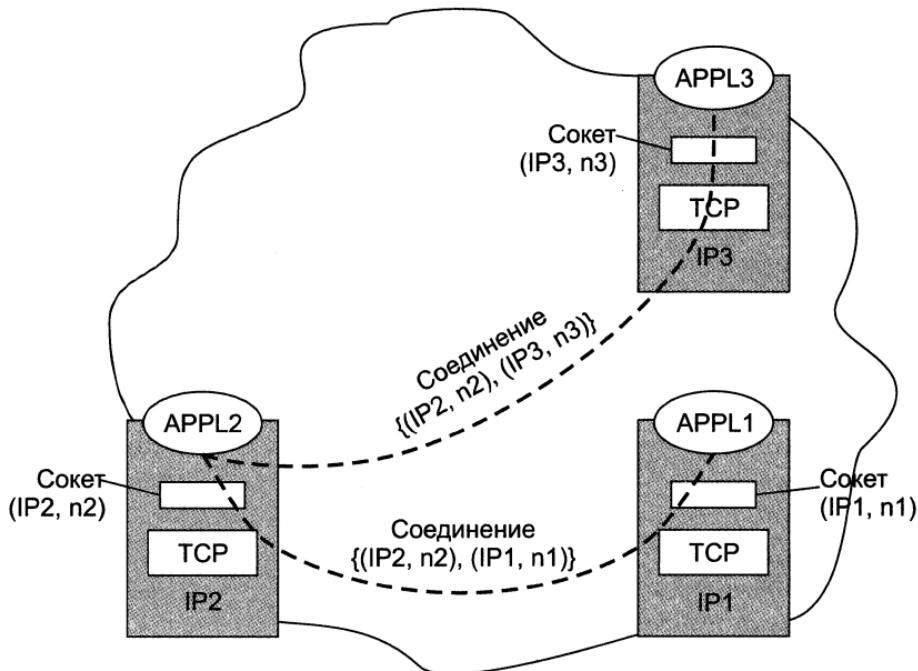


Рис. 3.25. Один сокет может участвовать в нескольких соединениях

На рисунке показаны два логических соединения, которое установило приложение 2 с приложением 1 и приложением 3. Логические соединения идентифицируются как $\{(IP2, n2), (IP1, n1)\}$ и $\{(IP2, n2), (IP3, n3)\}$ соответственно. Мы видим, что в обоих соединениях участвует один и тот же сокет — $(IP2, n2)$.

А 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. Веб-сервер основан на протоколе прикладного уровня HTTP, который передает свои сообщения в TCP-сегментах. TCP ожидает запросы от веб-клиентов (браузеров), «прослушивая» хорошо известный порт 80.

На рис. 3.26 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 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 можно сделать вполне конкретный вывод.

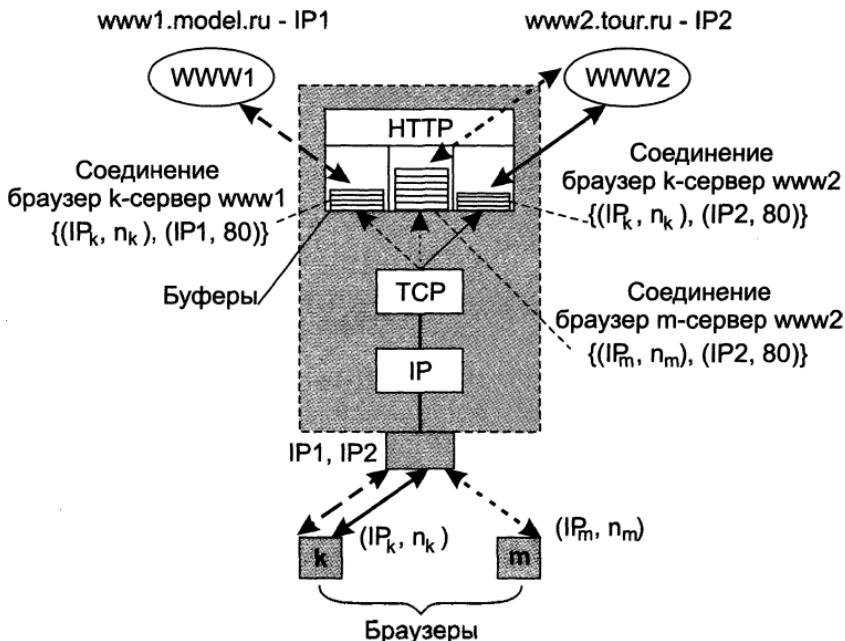


Рис. 3.26. Демультиплексирование протокола TCP на основе соединений

Протокол TCP осуществляет демультиплексирование информации, поступающей на прикладной уровень, на основе *соединений* процессов или, что одно и тоже, на основе идентифицирующих эти процессы *пар сокетов*.

Повторная передача и скользящее окно

Один из наиболее естественных приемов, используемых для организации надежной передачи — **квитирование**. Отправитель отсылает данные и ждет, пока к нему не придет квитанция, подтверждающая, что его данные благополучно дошли до адресата. В протоколе TCP используется частный случай квитирования — алгоритм скользящего окна. Прежде чем перейти к подробному рассмотрению особенностей реализации этого алгоритма в протоколе TCP, очень полезно обсудить этот метод с общих позиций.

Итак, существуют два метода организации процесса обмена квитанциями: метод простого источника и метод скользящего окна.

Метод простого источника требует, чтобы источник, пославший кадр (в данном случае не имеет значение, какое название используется для единицы передаваемых данных), дождался от приемника квитанции, извещающей о том, что исходный кадр получен и данные в нем корректны, и только *после этого* посыпал следующий кадр (или повторял искаженный). Если же квитанция в течение тайм-аута не пришла, то кадр (или квитанция) считается утерянным и его передача повторяется. На рис. 3.27 показано, что второй кадр отсылается только после того, как пришла квитанция, подтверждающая доставку первого кадра. Однако затем произошла длительная пауза в отправке следующего третьего кадра. В течение этой паузы источник был вынужден повторить передачу кадра 2, так как квитанция на первую его копию была потеряна. Понятно, что при таком алгоритме работы источника принимающая сторона должна уметь распознавать и избавляться от дублирующихся кадров.

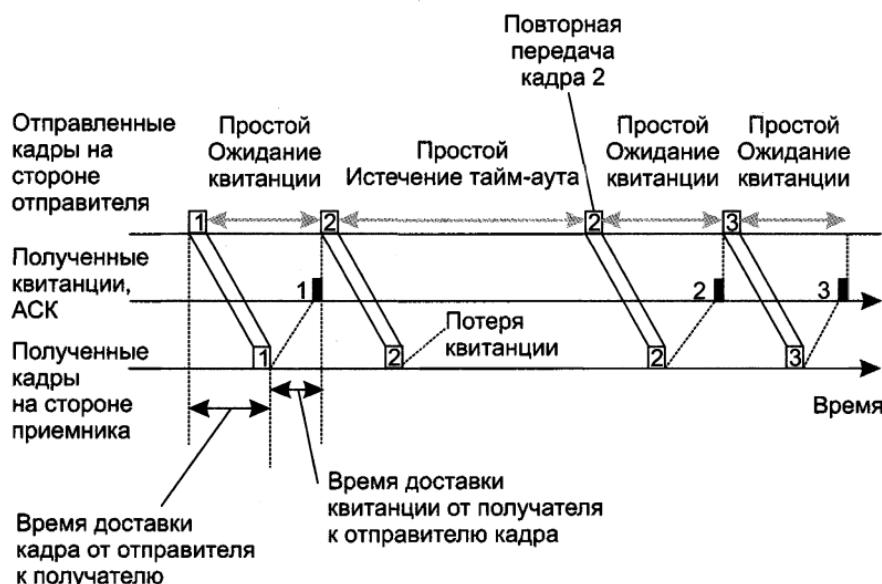


Рис. 3.27. Метод простого источника

Достаточно очевидно, что, что при использовании данного метода производительность обмена данными ниже потенциально возможной — передатчик мог бы посылать следующий кадр сразу же после отправки предыдущего, но он обязан ждать прихода квитанции.

Второй метод называется методом **скользящего окна** (sliding window). В этом методе для повышения скорости передачи данных источнику разрешается передать некоторое количество кадров в непрерывном режиме, то есть в максимально возможном для источника темпе еще *до получения на эти кадры квитанций*. Количество кадров, которые разрешается передавать таким образом, называется **размером окна**.

Рисунок 2.28 иллюстрирует применение данного метода для окна размером 5 кадров.

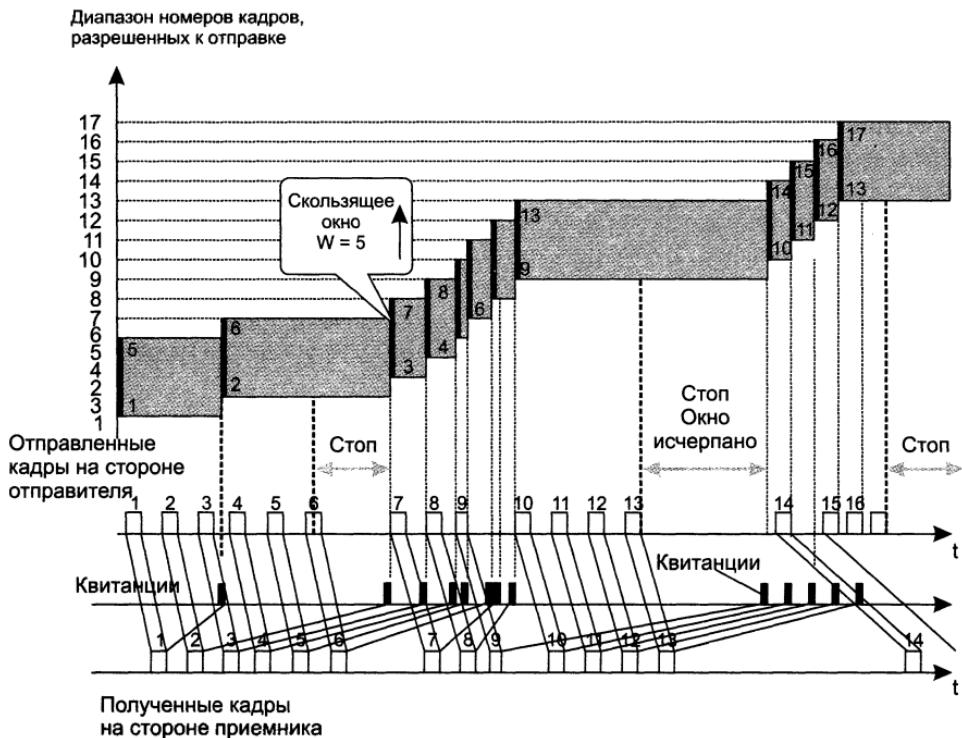


Рис. 3.28. Метод скользящего окна

В начальный момент, когда еще не послано ни одного кадра, окно определяет диапазон номеров кадров от 1 до 5 включительно. Источник начинает передавать кадры и через какое-то время получать в ответ квитанции. Для простоты предположим, что квитанции поступают в той же последовательности (но не обязательно в том же темпе), что и кадры, которым они соответствуют. В момент получения отправителем квитанции 1 окно сдвигается

на одну позицию вверх, определяя новый диапазон разрешенных к отправке кадров (от 2 до 6).

Процессы отправки пакетов и получения квитанций идут достаточно независимо друг от друга. В нашем примере отправитель продолжает передавать кадры, но некоторое время не получает на них квитанции. После передачи кадра 6 окно исчерпывается, и источник приостанавливает передачу.

После получения квитанции 2 (на кадр 2) окно сдвигается вверх на единицу, определяя диапазон разрешенных к передаче кадров 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 и остается равным 5. 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 кадров. Отправив последний разрешенный кадр 13, передатчик снова прекращает передачу с тем, чтобы возобновить ее после прихода квитанции 9.

При отправке кадра в источнике устанавливается таймаут. Если за установленное время квитанция на отправленный кадр не придет, то кадр (или квитанция на него) считается утерянным, и кадр передается снова. Если же поток квитанций поступает регулярно в пределах допуска в 5 кадров, то скорость обмена достигает максимально возможной величины для данного канала и принятого протокола.

В общем случае метод скользящего окна более сложен в реализации, чем метод простого источника, так как передатчик должен хранить в буфере копии всех кадров, на которые пока не получены квитанции. Кроме того, при использовании данного метода требуется отслеживать несколько параметров алгоритма, таких как размер окна, номер кадра, на который получена квитанция, номер кадра, который еще можно передать до получения новой квитанции.

Реализация метода скользящего окна в протоколе TCP

Алгоритм скользящего окна в протоколе TCP имеет некоторые существенные особенности. В частности, в рассмотренном ранее обобщенном алгоритме скользящего окна единицей передаваемых данных является кадр, и размер окна также определяется в кадрах, в то время как в протоколе TCP дело обстоит совсем по-другому.

Хотя единицей передаваемых данных протокола TCP является сегмент (аналог кадра в данном контексте), окно определено на множествеnumерованных байтов неструктурированного потока данных, передаваемого приложением протоколу TCP.

В ходе переговорного процесса модули TCP обоих 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) размёров 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. 3.29).

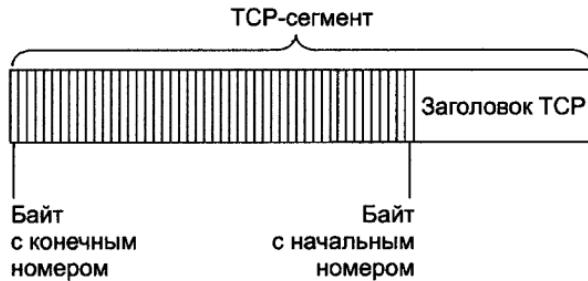


Рис. 3.29. Нумерация байтов в TCP-сегменте

Когда отправитель посыпает TCP-сегмент, он помещает в поле *последовательного номера* номер первого байта данного сегмента, который служит *идентификатором* сегмента. На рис. 3.30 показаны четыре сегмента размером 1460 байт и один — 870 байт. Идентификаторами этих сегментов являются номера 32600, 34060, 35520 и т. д. На основании этих номеров TCP-получатель не только отличает данный сегмент от других, но и позиционирует полученный фрагмент относительно общего потока байтов. Кроме того, он может сделать вывод, например, что полученный сегмент является дубликатом или между двумя полученными сегментами пропущены данные и т. д.

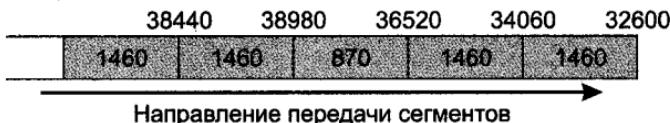


Рис. 3.30. Порядковый номер и номер квитанции

В качестве квитанции получатель сегмента отсылает ответное сообщение (сегмент), в поле *подтвержденного номера* которого он помещает число, на единицу превышающее максимальный номер байта в полученном сегменте. Так, для первого отправленного сегмента, изображенного на рис. 3.30, квитанцией о получении (подтвержденным номером) будет число 34 060, для второго — 35 520 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 посыпается только в случае правильного приема данных. 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.

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. 3.31).

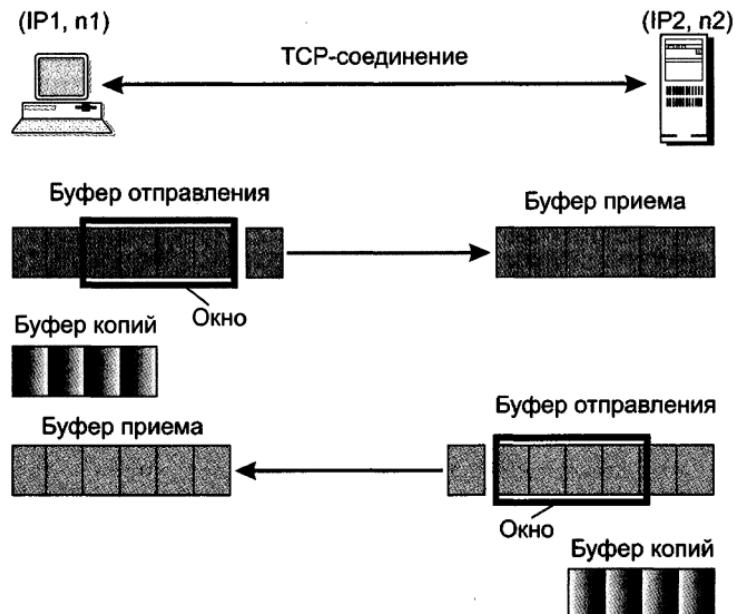


Рис. 3.31. Система буферов TCP-соединения

И при установлении соединения и в ходе передачи обе стороны, выступая в роли получателя, посыпают друг другу так называемые **окна приема**. Каждая из сторон, получив окно приема, «узнает», сколько байтов ей разрешается отправить с момента получения последней квитанции. Другими словами, посыпая 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.

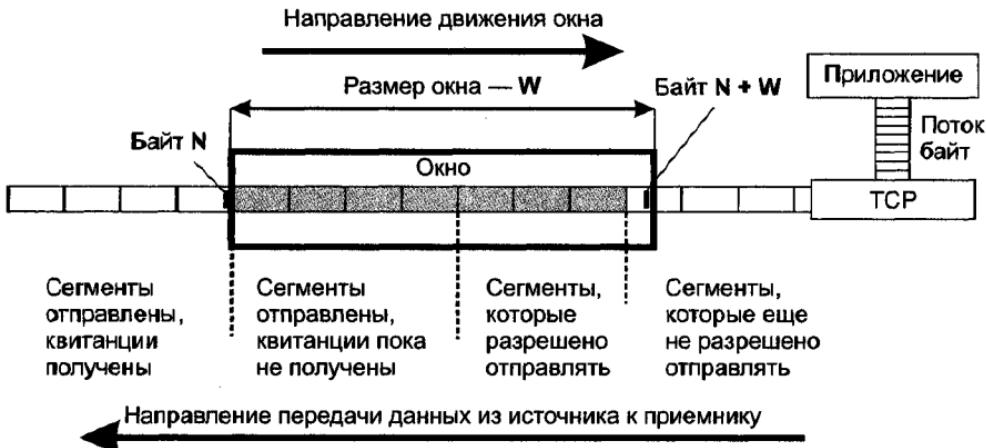


Рис. 3.32. Особенности реализации алгоритма скользящего окна в протоколе TCP

На рис. 3.32 показан поток байтов, поступающий от приложения в выходной буфер протокола TCP. Из потока байтов модуль TCP «нарезает» последовательность сегментов и поочередно отправляет их приложению-получателю. Для определенности на рисунке принято направление передачи 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.

Если размер окна равен W , а последняя по времени квитанция содержала значение N , то отправитель может посыпал 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. 3.33, 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*.

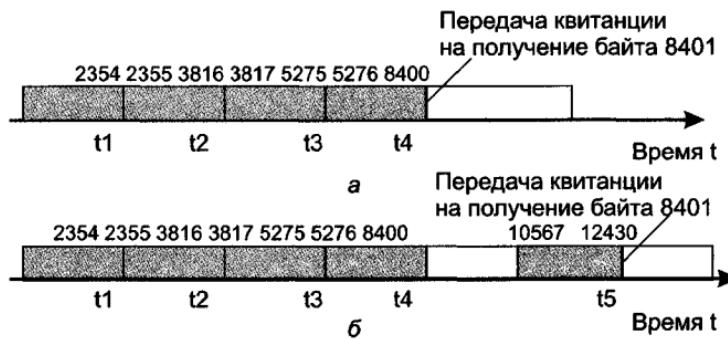


Рис. 3.33. Накопительный принцип квитирования:

- 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. 3.33, *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 бы означало, 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 подтверждая получение не отдельных блоков данных, а непрерывной последовательности байтов.

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, посыпается повторно. Может случиться так, что копия сегмента придет тогда, когда исходный сегмент уже окажется на месте, тогда дубликат будет попросту отброшен.

Управление потоком

Какой размер окна должен назначить источник приемнику, и наоборот? Точнее, какими величинами время ожидания (тайм-аут) очередной квитанции должно быть выбрано на каждой из сторон? От ответа на этот вопрос зависит производительность протокола TCP.

При выборе величины *тайм-аута* должны учитываться скорость и надежность линий связи, их протяженность и многие другие факторы. Тайм-аут 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.

В протоколе TCP тайм-аут определяется с помощью достаточно сложного *адаптивного* алгоритма, идея которого состоит в следующем. При каждой передаче засекается время от момента отправки сегмента до прихода квитанции о его приеме (время оборота). Получаемые значения времени оборота усредняются с весовыми коэффициентами, возрастающими от предыдущего замера к последующему. Это делается с тем, чтобы усилить влияние последних замеров. В качестве тайм-аута 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 выборе тайм-аута учитывается и дисперсия этой величины.

Размер окна приема связан с наличием в данный момент места в буфере данных у принимающей стороны. Поэтому 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 больше, чем клиент серверу. В зависимости от состояния сети то одна, то другая стороны могут объявлять новые значения окон приема, динамически уменьшая и увеличивая их.

Варьируя величину окна, можно влиять на загрузку сети. Чем больше окно, тем большая порция неподтвержденных данных может быть послана в сеть. Но если пришло большее количество данных, чем может быть принято модулем TCP, данные отбрасываются. Это ведет к излишним пересылкам информации и ненужному увеличению нагрузки на сеть и модуль TCP.

В то же время окно малого размера может ограничить передачу данных скоростью, которая определяется временем путешествия по сети каждого посылаемого сегмента. Чтобы избежать применения малых окон, в некоторых реализациях TCP предлагается получателю данных откладывать реальное изменение 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.

Управлять размером окна приема может не только та сторона, которая посыпает это окно, чтобы регулировать поток данных в свою сторону, но и вторая сторона — потенциальный отправитель данных. Если вторая сторона фиксирует ненадежную работу линии связи (регулярно запаздывают квитанции, часто требуется повторная передача), то она может по собственной инициативе уменьшить окно. В таких случаях действует правило: в качестве действующего размера окна выбирается минимальное из двух значений: значения, диктуемого приемной стороной, и значения, определяемого «на месте» отправителем.

Признаком перегрузки TCP-соединения является возникновение очередей на промежуточных узлах (маршрутизаторах) и на конечных узлах (компьютерах). При переполнении приемного буфера конечного узла «перегруженный» протокол TCP, отправляя квитанцию, помещает в нее новый уменьшенный размер окна. 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*. 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 посыпает квитанцию с указанием ненулевого размера окна.

Как видно из нашего далеко не полного описания двух протоколов транспортного уровня стека TCP/IP, на один из них — TCP — возложена сложная и очень важная задача обеспечения надежной передачи данных через ненадежную сеть.

В то же время функциональная простота протокола UDP обуславливает простоту алгоритма его работы, компактность и высокое быстродействие. Поэтому те приложения, в которых реализован собственный достаточно надежный механизм обмена сообщениями, основанный на установлении соединения, предпочитают для непосредственной передачи данных по сети использовать менее надежные, но более быстрые средства транспортировки, в качестве которых по отношению к протоколу TCP и выступает протокол UDP. Протокол UDP может применяться и в том случае, когда хорошее качество линий связи обеспечивает достаточный уровень надежности и без применения дополнительных приемов наподобие установления логического соединения и квитирования передаваемых пакетов. Заметим также, что поскольку протокол TCP основан на логических соединениях, он, в отличие от протокола UDP, *не годится для широковещательной и групповой рассылки*.

Протоколы маршрутизации

В этом разделе рассматриваются протоколы маршрутизации, которые обеспечивают поиск и фиксацию маршрутов продвижения данных через составную сеть TCP/IP. Но вначале мы остановимся на некоторых общих свойствах протоколов данного класса.

Общие свойства протоколов маршрутизации

Начнем с того, что существуют такие способы продвижения пакетов в составных сетях, которые вообще *не требуют наличия таблиц маршрутизации в маршрутизаторах*.

Наиболее простым способом передачи пакетов по сети является так называемая **лавинная маршрутизация**, когда 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, но он работоспособен (именно так мосты и коммутаторы локальных сетей поступают с кадрами, имеющими неизвестные адреса).

Еще одним видом маршрутизации, не требующим наличия таблиц маршрутизации, является **маршрутизация от источника** (source routing). В этом случае отправитель помещает в пакет информацию о том, какие промежуточные маршрутизаторы должны участвовать в передаче пакета к сети назначения. На основе этой информации каждый маршрутизатор считывает адрес следующего маршрутизатора, и если он действительно является адресом его непосредственного соседа, передает ему пакет для дальнейшей обработки. Вопрос о том, как отправитель узнает точный маршрут следования пакета через сеть, остается открытым. Маршрут может задавать либо вручную администратор, либо автоматически узел-отправитель, но в этом случае ему нужно поддерживать какой-либо протокол маршрутизации, который сообщает 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 маршрутизация от источника является одним из стандартных режимов продвижения пакетов, существует даже специальный заголовок для реализации этого режима.

Большинство же протоколов маршрутизации *нацелены на создание таблиц маршрутизации*.

Выбор рационального маршрута может осуществляться на основании различных **критерииев**. 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.

Различные протоколы маршрутизации обладают *разным временем конвергенции*.

Протокол маршрутизации должен создавать в маршрутизаторах *согласованные* друг с другом таблицы маршрутизации, то есть такие, которые обеспечивают доставку пакета от исходной сети в сеть назначения за конечное число шагов. Современные протоколы маршрутизации поддерживают согласованность таблиц, однако это их свойство не абсолютно — при изменениях в сети, например, при отказе каналов передачи данных или самих маршрутизаторов, возникают периоды нестабильной работы сети, вызванной временной несогласованностью таблиц разных маршрутизаторов. Протоколу маршрутизации обычно нужно некоторое время, которое называется *временем конвергенции*, чтобы после нескольких итераций обмена служебной информацией все маршрутизаторы сети внесли изменения в свои таблицы и в результате таблицы снова стали согласованными.

Различают протоколы, выполняющие статическую и адаптивную (динамическую) маршрутизацию.

При **статической маршрутизации** все записи в таблице имеют неизменяемый, статический 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.

При **адаптивной маршрутизации** все изменения конфигурации сети *автоматически* отражаются в таблицах маршрутизации благодаря *протоколам маршрутизации*. Эти протоколы собирают информацию о топологии связей в сети, что позволяет им оперативно отрабатывать 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, пакеты по нему больше не посылаются.

Протоколы адаптивной маршрутизации бывают распределенные и централизованные.

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. При *централизованном* подходе в сети существует один выделенный маршрутизатор, который собирает всю

информацию о топологии и состоянии сети **от** других маршрутизаторов. На основании этих данных выделенный маршрутизатор (который иногда называют *сервером маршрутов*) строит таблицы маршрутизации для всех остальных маршрутизаторов сети, а затем распространяет их по сети, чтобы каждый маршрутизатор получил собственную таблицу и в дальнейшем самостоятельно принимал решение о продвижении каждого пакета.

Применяемые сегодня в IP-сетях протоколы маршрутизации относятся к *адаптивным распределенным* протоколам, которые, в свою очередь, делятся на две группы:

- дистанционно-векторные алгоритмы (Distance Vector Algorithms, DVA);
- алгоритмы состояния связей (Link State Algorithms, LSA).

В **дистанционно-векторных алгоритмах** (DVA) каждый маршрутизатор *периодически* и *широковещательно* рассыпает по сети вектор, компонентами которого являются расстояния (измеренные в той или иной метрике) от данного маршрутизатора до всех известных ему сетей. Пакеты протоколов маршрутизации обычно называют *объявлениями о расстояниях*, так как с их помощью маршрутизатор объявляет остальным маршрутизаторам известные ему сведения о конфигурации сети.

Получив от некоторого соседа вектор расстояний (дистанций) до известных тому сетей, маршрутизатор наращивает компоненты вектора на величину расстояния от себя до данного соседа. Кроме того, он дополняет вектор информацией об известных ему самому других сетях, о которых он узнал непосредственно (если они подключены к его портам) или из аналогичных объявлений других маршрутизаторов. Обновленное значение вектора маршрутизатор рассыпает своим соседям. В конце концов, каждый маршрутизатор узнает через соседние маршрутизаторы информацию обо всех имеющихся в составной сети сетях и о расстояниях до них.

Затем он выбирает из нескольких альтернативных маршрутов к каждой сети тот маршрут, который обладает наименьшим значением метрики. Маршрутизатор, передавший информацию о данном маршруте, отмечается в таблице маршрутизации как *следующий* (next hop).

Дистанционно-векторные алгоритмы хорошо работают только в небольших сетях. В больших сетях они периодически засоряют линии связи интенсивным трафиком, к тому же изменения конфигурации не всегда корректно могут отрабатываться 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 RIP (Routing Information Protocol).

Алгоритмы состояния связей (LSA) обеспечивают каждый маршрутизатор информацией, достаточной для построения точного графа связей сети. Все маршрутизаторы работают на основании одного и того же графа, что делает процесс маршрутизации более устойчивым к изменениям конфигурации.

Каждый маршрутизатор использует граф сети для нахождения оптимальных по некоторому критерию маршрутов до каждой из сетей, входящих в составную сеть.

Чтобы понять, в каком состоянии находятся линии связи, подключенные к его портам, маршрутизатор периодически обменивается короткими пакетами HELLO со своими ближайшими соседями. В отличие от протоколов DVA, которые регулярно передают вектор расстояний, протоколы LSA 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.

В результате служебный трафик, создаваемый протоколами LSA, гораздо менее интенсивный, чем у протоколов DVA.

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.

Протокол OSPF

Рассмотрим более детально работу протокола маршрутизации на примере **протокола OSPF** (Open Shortest Path First – выбор кратчайшего пути первым). Как и все протоколы маршрутизации, основанные на алгоритме состояния связей, OSPF разбивает процедуру построения таблицы маршрутизации на две, к первой относится построение и поддержание базы данных о состоянии связей сети, ко второй – нахождение оптимальных маршрутов и генерация таблицы маршрутизации.

- ❑ *Построение и поддержание базы данных о состоянии связей сети.* Связи сети могут быть представлены в виде графа, в котором вершинами графа являются маршрутизаторы и подсети, а ребрами – связи между ними (рис. 3.34). Каждый маршрутизатор обменивается со своими соседями той информацией о графике сети, которой они располагают к данному моменту. Этот процесс похож на процесс распространения векторов расстояний до сетей в протоколе RIP, однако сама информация качественно иная – это информация о топологии сети. Сообщения, с помощью которых распространяется топологическая информация, называются **объявлениями о состоянии связей** сети (Link State Advertisements, LSA). 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.

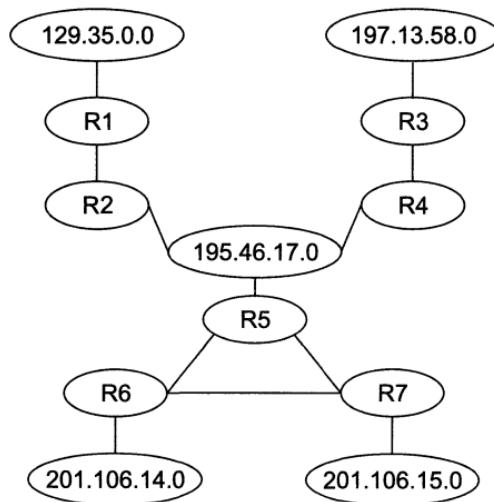


Рис. 3.34. Граф сети, построенный протоколом OSPF

Для контроля состояния связей и соседних маршрутизаторов OSPF-маршрутизаторы передают друг другу сообщения HELLO каждые 10 секунд. Небольшой объем этих сообщений делает возможным частое тестирование состояния соседей и связей с ними. В том случае, когда сообщения HELLO перестают поступать от какого-либо непосредственного соседа, маршрутизатор делает вывод о том, что состояние связи изменилось с работоспособного на неработоспособное и вносит соответствующие корректизы в свою топологическую базу данных. Одновременно он отсылает всем непосредственным соседям объявление об этом изменении, и те также вносят исправления в свои базы данных и, в свою очередь, рассылают данное объявление LSA своим непосредственным соседям.

- **Нахождение оптимальных маршрутов и генерация таблицы маршрутизации.** Задача нахождения оптимального пути на графике является достаточно сложной и трудоемкой. В протоколе OSPF для ее решения используется итеративный алгоритм Дийкстры. 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.

Если состояние связей в сети изменилось, и произошла корректировка графа сети, 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 с дистанционно-векторными протоколами. Это время состоит из времени распространения по сети объявления LSA и времени работы алгоритма Дийкстры, который обладает быстрой сходимостью. Однако большая вычислительная сложность этого алгоритма предъявляет высокие требования к мощности процессора маршрутизатора.

Когда 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 OSPF-маршрутизаторы 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.

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, минимизирующие 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 Т-1¹, обладающего пропускной способностью 1,544 Мбит/с, – 65, для канала с пропускной способностью 56 Кбит/с – 1785. При наличии высокоскоростных каналов, таких как Gigabit Ethernet или STM-16/64, администратору нужно задать другую шкалу скоростей, назначив единичное расстояние наиболее скоростному каналу.

¹ Т-1 – это цифровой канал технологии PDH, которая будет рассматриваться в главе 4.

При выборе оптимального пути на графике с каждым ребром графа связывается метрика, которая добавляется к пути, если данное ребро в него входит. Пусть в приведенном на рис. 3.34 примере маршрутизатор R5 связан с маршрутизаторами R6 и R7 каналами Т-1, а маршрутизаторы R6 и R7 связаны между собой каналом 56 Кбит/с. Тогда R7 определит оптимальный маршрут до сети 201.106.14.0 как составной, проходящий сначала через R5, а затем через R6, поскольку у этого маршрута метрика будет равна $65 + 65 = 130$ единиц. Непосредственный маршрут через R6 не будет оптимальным, так как его метрика равна 1785.

Протокол OSPF разрешает хранить в таблице маршрутизации несколько маршрутов к одной сети, если они обладают равными метриками. В таких случаях маршрутизатор может работать в режиме баланса загрузки маршрутов, отправляя пакеты попаременно по каждому из маршрутов.

К сожалению, вычислительная сложность протокола OSPF быстро растет с увеличением размера сети. Для преодоления этого недостатка в протоколе OSPF вводится понятие **области сети**. Маршрутизаторы, принадлежащие некоторой области, строят график связей только для этой области, что упрощает задачу. Между областями информация о связях не передается, а пограничные для областей маршрутизаторы обмениваются только информацией об адресах сетей, имеющихся в каждой из областей, и *расстоянием от пограничного маршрутизатора до каждой сети*. При передаче пакетов между областями выбирается один из пограничных маршрутизаторов области, а именно тот, у которого расстояние до нужной сети меньше.

Взаимодействие протоколов маршрутизации

В одной и той же сети могут одновременно работать несколько разных протоколов маршрутизации (рис. 3.35). Это означает, что на некоторых (не обязательно 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.

В маршрутизаторе, который поддерживает 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.

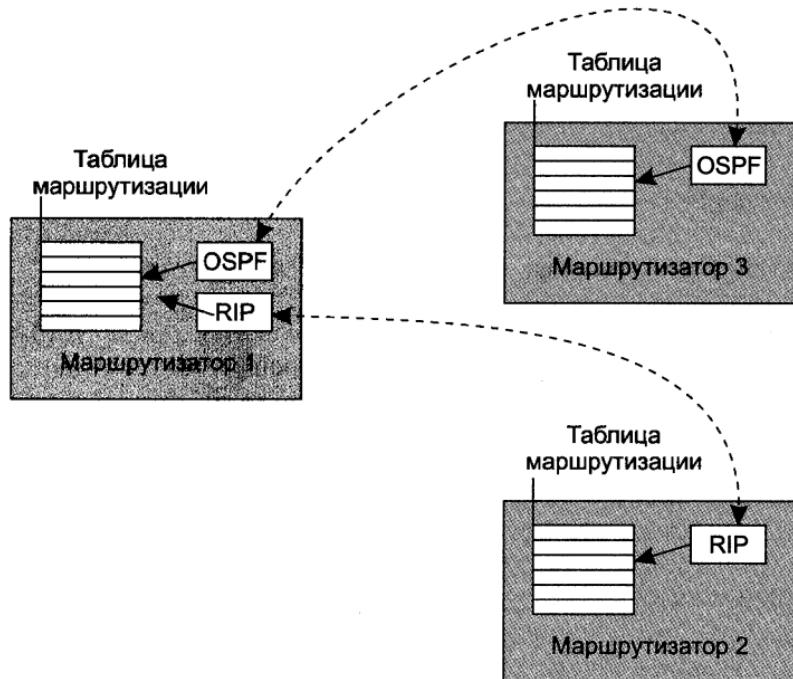


Рис. 3.35. Применение нескольких протоколов маршрутизации в одной сети

По умолчанию каждый протокол маршрутизации, работающий на определенном маршрутизаторе, распространяет только «собственную» информацию, то есть ту информацию, которая была получена данным маршрутизатором по данному протоколу. Например, если о маршруте к некоторой сети маршрутизатор узнал по протоколу RIP, то и распространять по сети объявления об этоммаршруте 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 решалась бы эффективнее, если бы маршрутизаторы могли обмениваться маршрутной информацией, полученной разными протоколами маршрутизации. Такая возможность реализуется в особом режиме работы маршрутизатора, называемом *перераспределением*. 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 не простое, от администратора требуется провести определенную работу по конфигурированию каждого маршрутизатора. Очевидно, что для крупных составных сетей нужно качественно иное решение.

Внешние и внутренние шлюзовые протоколы

Такое решение было найдено для самой крупной на сегодня составной сети — Интернета. Это решение базируется на понятии автономной системы.

Автономная система — 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 16 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, каждая из которых состоит из взаимосвязанных сетей (рис. 3.36), соединенными **внешними шлюзами**.

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 — теперь сначала маршрут определяется как *последовательность автономных систем*, затем — как *последовательность сетей* и только потом ведет к конечному узлу.

Выбор *маршрута между автономными системами* осуществляют внешние шлюзы, использующие особый тип протокола маршрутизации, так называемый **внешний шлюзовой протокол** (Exterior Gateway Protocol, EGP). В настоящее время для работы в такой роли сообщество Интернет 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 Protocols, IGP). К числу протоколов IGP относятся знакомые нам RIP, OSPF, IS-IS. В случае транзитной автономной системы эти протоколы указывают точную последовательность маршрутизаторов от точки входа в автономную систему до точки выхода из нее.

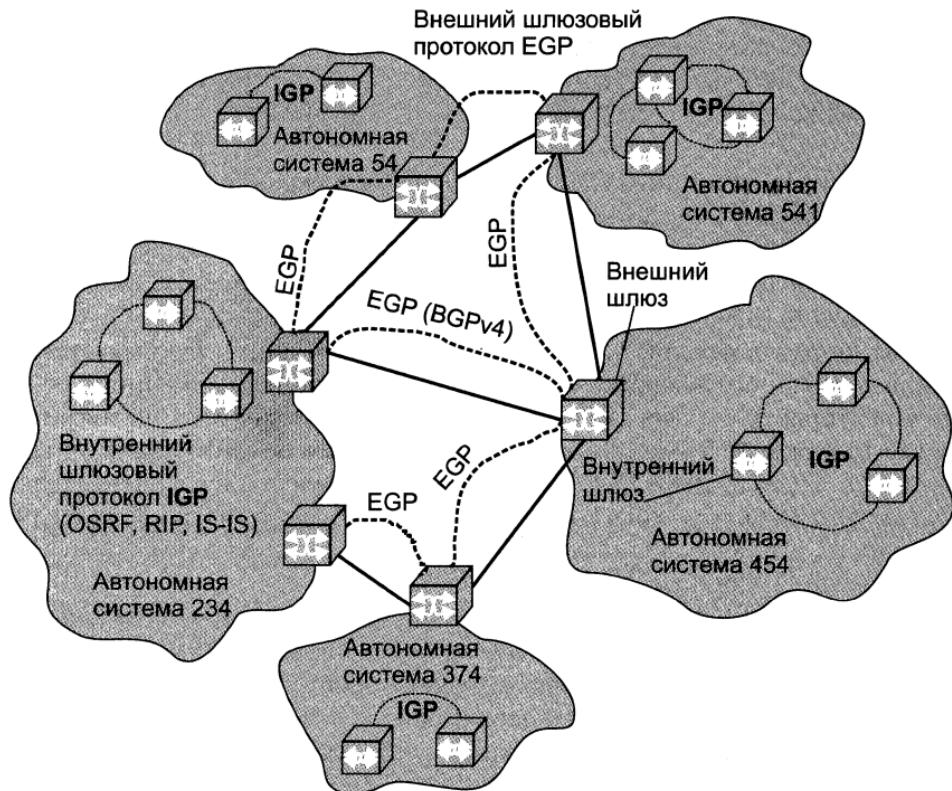


Рис. 3.36. Автономные системы Интернета

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.

Автономные системы составляют **магистраль** Интернета. Концепция автономных систем скрывает от администраторов магистрали Интернета проблемы маршрутизации пакетов на более низком уровне — уровне сетей. Для администратора магистрали неважно, какие протоколы маршрутизации применяются внутри автономных систем, для него существует единственный протокол маршрутизации — **BGPv4**.

Вспомогательные протоколы и средства стека TCP/IP

Протокол ICMP

Протокол межсетевых управляющих сообщений (Internet Control Message Protocol, ICMP) (RFC 792) является вспомогательным протоколом, использующимся для диагностики и мониторинга сети.

Существует ряд ситуаций, когда протокол IP не может доставить пакет адресату, например, когда истекает время жизни пакета, когда в таблице маршрутизации отсутствует маршрут к заданному в пакете адресу назначения, когда пакет не проходит проверку по контрольной сумме, когда шлюз не имеет достаточно места в своем буфере для передачи какого-либо пакета и т. д., и т. п. Как мы не раз отмечали, протокол IP доставляет данные, руководствуясь принципом «по возможности» (best efforts), то есть не предпринимает мер для гарантированной передачи данных адресату. Это свойство «необязательности» протокола IP 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 служит дополнением, компенсирующим ненадежность протокола IP, но несколько *другого рода*. Он не предназначен для исправления возникших при передаче пакета проблем: если пакет потерян, 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). Протокол IP, обнаруживший, что он не может передать IP-пакет далее по сети, прежде чем отбросить пакет, должен отправить *диагностическое ICMP-сообщение* конечному узлу-источнику. Для передачи по сети ICMP-сообщение инкапсулируется в поле данных IP-пакета. IP-адрес узла-источника определяется из заголовка пакета, вызвавшего инцидент.

Сообщение, прибывшее в узел-источник, может быть обработано там либо ядром операционной системы, либо протоколами транспортного и прикладного уровней, либо приложениями, либо просто проигнорировано. Важно, что обработка ICMP-сообщений не входит в обязанности протоколов IP и ICMP.

Заметим, что некоторые из пакетов могут исчезнуть в сети, не вызвав при этом никаких оповещений. В частности, протокол ICMP не предусматривает

передачу сообщений о проблемах, возникающих при обработке IP-пакетов, несущих ICMP-сообщения об ошибках. Такое решение было принято разработчиками протокола, чтобы не порождать «штормы» в сетях, когда количество сообщений об ошибках лавинообразно возрастает.

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. 3.37), однако интерпретация полей существенно зависит от того, к какому типу относится сообщение.

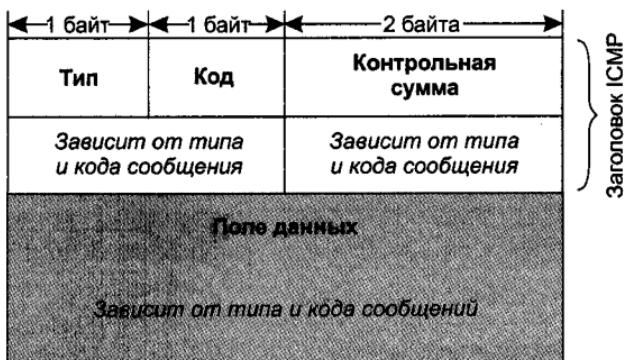


Рис. 3.37. Формат ICMP-сообщения

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 четырех байтов в заголовке и поле данных зависит от значений полей типа и кода.

На рис. 3.38 показана таблица основных типов ICMP-сообщений. Эти сообщения можно разделить на две группы (помеченные на рисунке условными символами):

- сообщения об ошибках;
- сообщения запрос-ответ.

Сообщения типа запрос-ответ связаны в пары: эхо-запрос – эхо-ответ, запрос маски – ответ маски, запрос времени – ответ времени. Отправитель сообщения-запроса всегда рассчитывает на получение соответствующего сообщения-ответа.

Сообщения, относящиеся к группе сообщений об ошибке, конкретизируются уточняющим кодом. На рисунке показан фрагмент таблицы кодов для сообщения об ошибке недостижимости узла назначения, имеющей тип 3.

Из таблицы мы видим, что это сообщение может быть вызвано различными причинами, такими как неверный адрес сети или узла (код 0 или 1), отсутствием 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 ошибке.

Таблица типов ICMP-сообщений

Значение в поле "Тип"	Тип сообщения
0	Эхо-ответ
3	Узел назначения недостижим
4	Подавление источника
5	Перенаправление маршрута
8	Эхо-запрос
11	Истечение времени диаграммы
12	Проблема с параметрами пакета
13	Запрос отметки времени
14	Ответ отметки времени
17	Запрос маски
18	Ответ маски

Таблица кодов причин ошибок 3

Код	Причина
0	Сеть не достижима
1	Узел не достижим
2	Протокол не достижим
3	Порт не достижим
4	Ошибка фрагментации
5	Ошибка в маршруте источника
6	Сеть назначения не известна
7	Узел назначения не известен
8	Узел-источник изолирован
9	Административный запрет

? сообщение-запрос

i сообщение-ответ

✓ сообщение-ошибка

Рис. 3.38. Типы и коды ICMP-сообщений

Утилита traceroute

В качестве примера рассмотрим использование сообщений об ошибке для построения популярной утилиты мониторинга сети `traceroute`.

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. 3.39. 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.

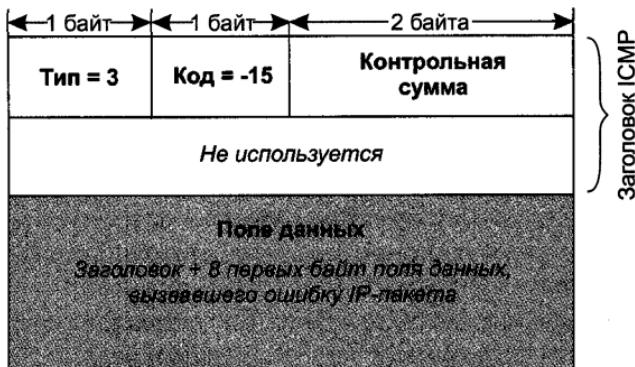


Рис. 3.39. Формат ICMP-сообщения об ошибке недостижимости узла назначения

Помимо причины ошибки, указанной в заголовке (в полях типа и кода), дополнительная диагностическая информация передается в поле данных ICMP-сообщения. Именно туда помещается заголовок IP и *первые 8 байт данных* того IP-пакета, который вызвал ошибку. Эта информация позволяет узлу-правителю еще более точно диагностировать причину ошибки. Это возможно, так как все протоколы стека TCP/IP, использующие для передачи своих сообщений IP-пакеты, помещают наиболее важную для анализа информацию в первые 8 байт своих сообщений. В частности, ими вполне могут оказаться первые 8 байт заголовка TCP или UDP, в которых содержится информация (номер порта), идентифицирующая приложение, пославшее потерянный пакет. Следовательно, при разработке приложения можно предусмотреть встроенные средства реакции на сообщения о недоставленных пакетах.

ICMP-сообщения об ошибке лежат в основе работы популярной утилиты `traceroute` для UNIX, имеющей в Windows 2000 название `tracert`. Эта утилита позволяет проследить маршрут до удаленного хоста, определить среднее время оборота (RTT), IP-адрес и в некоторых случаях доменное имя каждого промежуточного маршрутизатора. Такая информация помогает найти маршрутизатор, на котором обрывается 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, то 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 посыпает следующий IP-пакет, но теперь со значением TTL, равным 2. Этот пакет благополучно проходит первый маршрутизатор, но «умирает» на втором, о чём 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.

Утилита ping

А сейчас давайте рассмотрим представителей другой группы ICMP-сообщений – **эхо-запросы** и **эхо-ответы** и поговорим об использовании этих сообщений в известной утилите ping.

Эхо-запрос и эхо-ответ, в совокупности называемые **эхо-протоколом**, представляют собой очень простое средство мониторинга сети. Компьютер или маршрутизатор посыпает 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, то их успешная доставка означает нормальное функционирование всей транспортной системы составной сети.

Формат эхо-запроса и эхо-ответа показан на рис. 3.40. Поле типа для эхо-ответа равно 0, 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. 3.40. 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, которая предназначена 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, которое послало этот запрос. А порядковый номер используется приложением, чтобы связать полученный ответ с соответствующим запросом (учитывая, что одно приложение может выдать несколько однотипных запросов).

На основе эхо-протокола функционирует весьма популярная утилита `ping`, предназначенная для тестирования достижимости узлов. Эта утилита обычно посыпает 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 приведенной распечатки видно, что в ответ на тестирующие запросы, посланные узлу `server1.mgu.ru`, было получено 4 эхо-ответа. Длина каждого сообщения составляет 64 байта. В следующей колонке помещены значения времени оборота (RTT), то есть времени от момента отправки запроса до получения ответа на этот 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.

Протокол NAT

Маршрутизация в составной сети осуществляется на основе тех адресов назначения, которые помещены в заголовки пакетов. Как правило, эти адреса остаются неизменными с момента их формирования отправителем до момента поступления на узел получателя. Однако из этого правила есть исключения. Например, широко применяемая сегодня технология **трансляции сетевых адресов** (Network Address Translation, NAT) позволяет пакетам перемещаться во внешней сети (в Интернете) на основании адресов, отличающихся от адресов маршрутизации пакета во частной (корпоративной) сети.

Одной из причин использования технологии NAT является *дефицит IP-адресов*. Если по каким-либо причинам предприятию, у которого имеется

потребность подключения к Интернету, не удается получить у поставщика услуг необходимого количества глобальных IP-адресов, то оно может задействовать для адресации внутренних узлов **частные адреса**¹, а затем привлечь технологию NAT, которая даст возможность узлам с частными адресами связываться между собой, а также через Интернет с узлами, имеющими глобальные адреса.

Технология NAT может быть также 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.

Технология трансляции сетевых адресов имеет несколько разновидностей, наиболее популярная из которых — **традиционная технология NAT** — позволяет узлам из частной сети прозрачным для пользователей образом получать доступ к узлам внешних сетей. Подчеркнем, что в данном варианте NAT решает проблему 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 частной сети, то сеанс называется исходящим, несмотря на то, что в его рамках в сеть могут поступать данные извне².

Идея традиционной технологии NAT состоит в следующем. Пусть сеть предприятия образует тупиковый домен, узлам которого присвоены частные адреса (рис. 3.41). На маршрутизаторе, связывающем сеть предприятия с внешней сетью, установлено программное обеспечение NAT, что превращает этот маршрутизатор в NAT-устройство. Это NAT-устройство динамически отображает набор частных адресов {IP*} на набор глобальных адресов {IP}, полученных предприятием от поставщика услуг и присвоенных внешнему интерфейсу маршрутизатора предприятия.

Важным для работы NAT-устройства является правило распространения маршрутных объявлений через границы частных сетей. Объявления протоколов маршрутизации о внешних сетях «пропускаются» пограничными маршрутизаторами в частные сети и обрабатываются внутренними маршрутизаторами. Обратное утверждение неверно — маршрутизаторы внешних сетей не получают объявлений о частных 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 две технологии:

¹ О частных адресах рассказывается в начале этой главы.

² 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.

- **базовая трансляция сетевых адресов** (Basic Network Address Translation, Basic NAT);
- **трансляция сетевых адресов и портов** (Network Address Port Translation, NAPT).

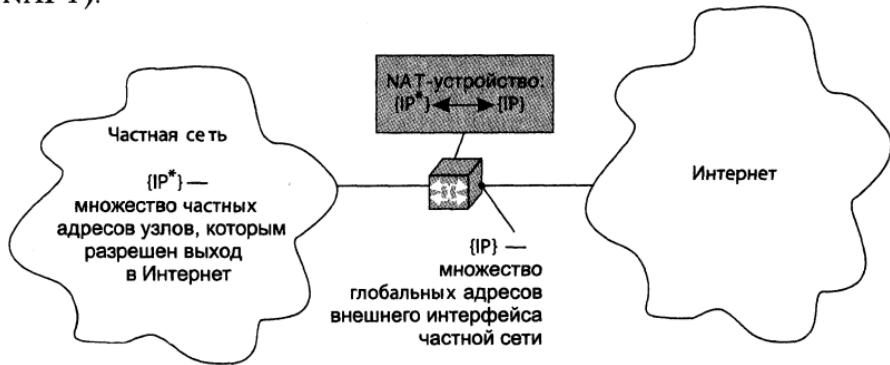


Рис. 3.41. Схема действия традиционной технологии NAT

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 TCP- и UDP-порты.

Начнем с *базовой трансляции сетевых адресов*. Если количество локальных узлов, которым необходимо обеспечить выход во внешнюю сеть, меньше или равно имеющегося количества 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.

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, брандмауэром), на котором установлено программное обеспечение NAT.

В разных тупиковых доменах могут быть совпадающие частные адреса. Например, в сетях A и B на рис. 3.42 для внутренней адресации применяется один и тот же блок частных 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 в сети B) являются глобальными. 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 маршрутизаторов R2 и R3 каждой из сетей.

Когда узел 10.0.1.4 сети A посыпает пакет хосту 10.0.1.2 сети B, то он помещает в заголовок пакета в качестве адреса назначения глобальный адрес

185.127.125.3/24. Узел-источник направляет пакет своему маршрутизатору R1 по умолчанию, 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 протокол 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.

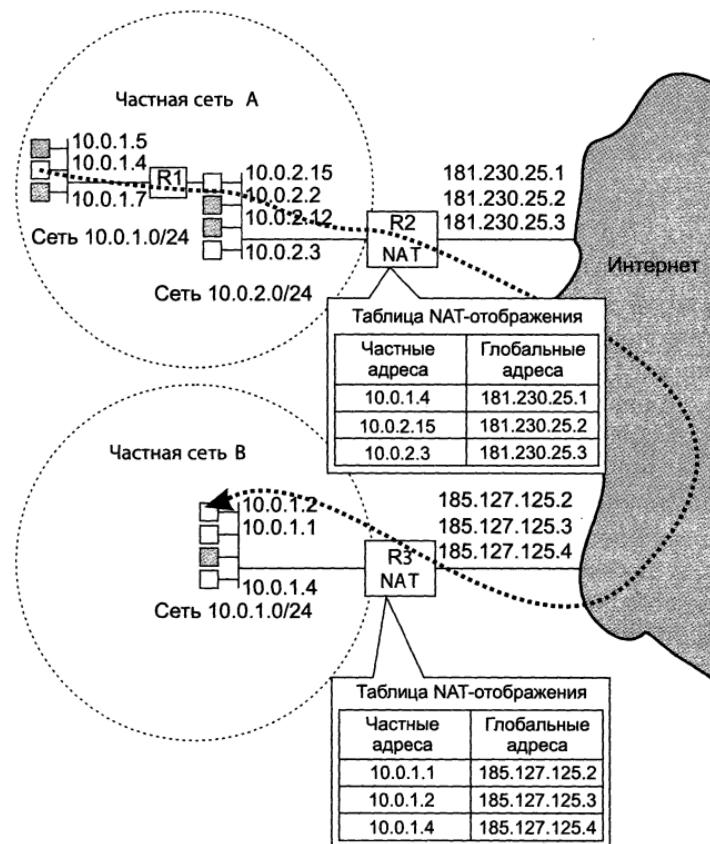


Рис. 3.42. Базовая трансляция сетевых адресов для исходящих сеансов

Заметим, что в описанной операции не требуется участия узлов отправителя и получателя, то есть она прозрачна для пользователей.

Теперь поговорим о *трансляции 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, а для адресации остальных узлов сети задействованы частные адреса. NAPT позволяет *всем* узлам внутренней сети одновременно взаимодействовать с внешними сетями через единственный зарегистрированный IP-адрес.

Возникает вопрос, каким образом пакеты, поступающие из Интернета в *ответ* на запросы из частной сети, находят узел, с которым они вступили в диалог? Ведь в поле адреса отправителя любого пакета, направленного по любому внешнему адресу, помещен один и тот же адрес – адрес внешнего интерфейса пограничного маршрутизатора.

Казалось бы, решение лежит на поверхности. Для однозначной идентификации узла отправителя нужно привлечь дополнительную информацию, такую, например, как номер UDP- или TCP-порта. Однако не все так просто, поскольку из частной 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 частной во внешнюю сеть каждой паре (частный IP-адрес отправителя; номер TCP/UDP-порта отправителя) ставится в соответствие пара (глобальный IP-адрес внешнего интерфейса; назначенный номер TCP- или UDP-порта). 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 с единственным зарегистрированным глобальным адресом хорошо работает в частных сетях средних размеров.

На рис. 3.43 приведен пример, когда в частной сети используются адреса из блока 10.0.0.0. Внешнему интерфейсу маршрутизатора этой сети поставщиком услуг назначен адрес 181.230.25.1.

Когда хост 10.0.1.4 внутренней сети посыпает во внешнюю сеть пакет серверу telnet, то он в качестве адреса назначения указывает 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 TCP-порт 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 сервер генерирует ответное сообщение, то он в качестве адреса назначения указывает единственный зарегистрированный глобальный адрес внутренней сети, являющийся адресом внешнего интерфейса NAPT-устройства. В качестве 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.

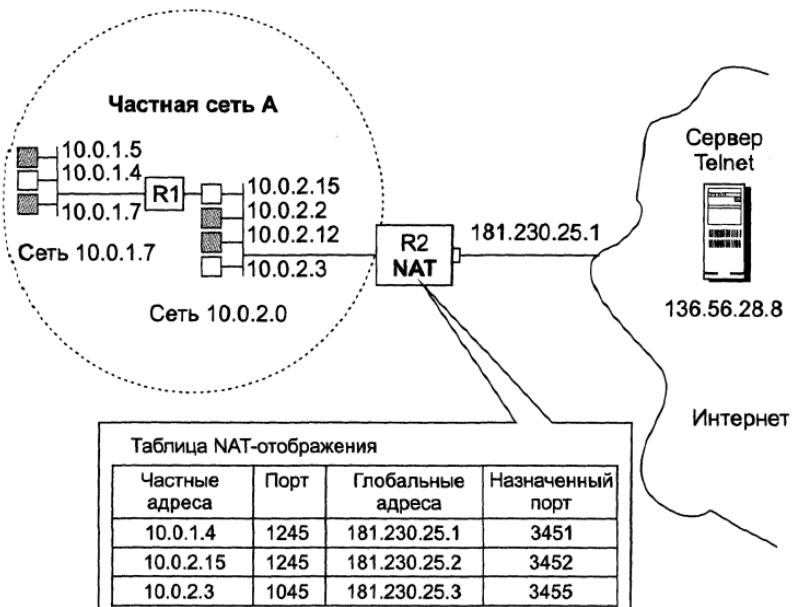


Рис. 3.43. Трансляция сетевых адресов и портов для исходящих TCP- и UDP-сеансов

ВНИМАНИЕ

Заметьте, что в таблице имеется еще одна запись с номером порта 1245, такая ситуация вполне возможна: операционные системы на разных компьютерах независимо присваивают номера портов клиентским программам. Именно для разрешения такой неоднозначности и привлекаются уникальные назначенные номера портов.

В технологии NAPT разрешаются только исходящие из частной сети TCP- и UDP-сеансы. Однако в тех случаях, когда служба и узел, на котором она работает, однозначно определяются хорошо известным зарегистрированным номером порта службы, можно обеспечить доступ к некоторому серверу внутренней сети извне.

Завершая рассмотрение технологии NAT, заметим, что помимо традиционной технологии NAT существуют и другие ее варианты, например, двойной трансляции сетевых адресов, при которой модифицируются оба адреса: и источника, и приемника (в отличие от традиционной технологии NAT, когда модифицируется только один адрес). Двойная трансляция сетевых адресов необходима при коллизиях частных и внешних адресных пространств. Наиболее часто это происходит, когда внутренний домен имеет некорректно назначенные публичные адреса, которые принадлежат другой организации. Подобная ситуация может возникнуть из-за того, что сеть организации была изначально изолированной и адреса назначались произвольно, причем из глобального пространства.

ГЛАВА 4. Технологии глобальных сетей

Самый отдаленный пункт земного шара
к чему-нибудь да близок, а самый близкий
от чего-нибудь да удален.

Козьма Протков

Сети IP, рассмотренные в предыдущей главе, могут быть как локальными, так и глобальными, сама технология IP не делает различия между этими двумя типами компьютерных сетей, отличающихся расстояниями между узлами сети. Наиболее известный представитель сетей IP – сеть Интернет – является глобальной сетью, а локальные сети IP вы можете встретить на любом предприятии.

В то же время существуют технологии компьютерных сетей, предназначенные специально для создания глобальных сетей: Frame Relay, ATM, MPLS. Сети, построенные на этих технологиях, могут покрывать большие территории и объединять большое количество узлов, оставаясь составляющими сетями единой сети IP. В этой главе мы рассмотрим особенности таких технологий, а также изучим принципы работы первичных сетей, служащих средой для создания коммуникационных каналов.

Первичные сети

Первичные, или транспортные, сети (*transmission networks*) – это телекоммуникационные сети особого вида, предназначенные для создания постоянных глобальных высокоскоростных каналов, которые затем используются для построения других сетей, например, телефонных или компьютерных.

Отличие первичных сетей от других телекоммуникационных сетей состоит в том, что они не работают с терминальными устройствами конечных пользователей, как это делают те же телефонные сети, связывающие телефонные аппараты, или компьютерные сети, соединяющие между собой компьютеры. Вместо этого каналы первичных сетей соединяют коммутационные

устройства других сетей, а уже те, в свою очередь, обслуживают терминалы конечных пользователей.

По отношению к первичным сетям телефонные и компьютерные сети являются **вторичными, или наложенными (overlay), сетями**.

Архитектура первичной сети соответствует обобщенной архитектуре телекоммуникационной сети (см. главу 1), то есть состоит из кабельных линий связи и коммутаторов.

В первичных сетях используется техника коммутации каналов, поэтому каналы этих сетей обладают *фиксированной пропускной способностью*.

Особенностью коммутаторов первичных сетей является то, что они коммутируют каналы не *динамически* по запросам пользовательских устройств, как это происходит в телефонных сетях, где набор номера на аппарате вызывает коммутацию составного канала с аппаратом вызываемого абонента, а *статически* по командам оператора сети.

Поэтому для двух коммутаторов наложенной сети соединяющих их составной канал первичной сети представляется простым постоянным кабельным соединением, коммутаторы наложенной сети «не видят» расположенных между ними коммутаторов первичной сети. В таких случаях говорят, первичная сеть *прозрачна* для работающей через ее каналы наложенной сети.

На рис. 4.1. показан фрагмент наложенной сети с коммутацией пакетов (верхняя плоскость), три маршрутизатора которой соединены через первичную сеть (нижняя плоскость).

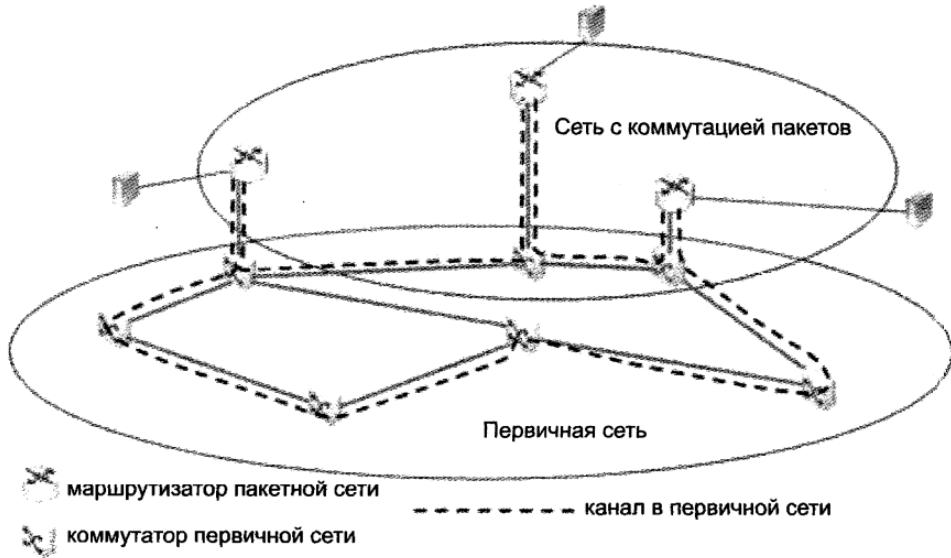


Рис. 4.1. Соединение маршрутизаторов через первичную сеть

Обычно один кабель первичной сети позволяет за счет мультиплексирования передавать трафик нескольких сотен магистральных каналов компьютерных или телефонных сетей.

Существует несколько технологий, используемых для создания первичных сетей:

- плезиохронная цифровая иерархия (PDH);
- синхронная цифровая иерархия (SDH / SONET);
- уплотненное волновое мультиплексирование (DWDM);
- оптические транспортные сети (OTN).

Сети PDH

Технология **плезиохронной цифровой иерархии** (Plesiochronous Digital Hierarchy, **PDH**) была разработана в конце 60-х годов компанией AT&T для связи крупных коммутаторов телефонных сетей между собой. Применяемые до этого аналоговые линии связи с частотным мультиплексированием исчерпали свои возможности по организации высокоскоростной и высококачественной многоканальной связи по одному кабелю. Переход на технологию PDH ознаменовал начало новой эпохи в телекоммуникационных и телефонных сетях — эпохи **цифровых** коммуникаций. Для конечных абонентов это означало высокое качество звука (см. далее раздел «Технология ATM»), которое не ухудшалось по мере прохождения промежуточных коммутаторов, как это происходило в аналоговых сетях. Для операторов это означало появление гибких средств создания надежных каналов с широким диапазоном скоростей, от единиц до сотен мегабитов в секунду.

Начало технологии PDH было положено разработкой мультиплексора **T-1**, 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 T-1 сами осуществляли оцифровывание голоса с частотой 8000 Гц, создавая абонентские элементарные цифровые каналы со скоростью передачи данных 64 Кбит/с.

В оборудовании T-1 используется техника синхронного временного мультиплексирования.

Временное мультиплексирование

Принцип **временного мультиплексирования** (Time Division Multiplexing, **TDM**), заключающийся в выделении канала каждому соединению на определенный период времени, используется во многих технологиях. Имеется два типа временного 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, каждый пользователь пытается занять канал тогда, когда у него возникает потребность в передаче информации.

Синхронный режим TDM¹ находит применение в сетях с *коммутацией каналов*, к которым относятся и сети PDH. В этом случае доступ всех информационных потоков к каналу синхронизируется таким образом, чтобы каждый информационный поток периодически получал канал в свое распоряжение на фиксированный промежуток времени.

Синхронизация оборудования TDM позволяет использовать временное положение кадра в цикле работы оборудования в качестве его адреса назначения — в этом состоит принципиальное отличие сетей TDM от сетей с коммутацией пакетов, где адрес назначения в кадре необходимо указывать явно.

Работу оборудования T-1 на основе этой техники иллюстрирует рис. 4.2, на котором показан фрагмент сети, состоящий из двух мультиплексоров² (*M1* и *M2*) и одного коммутатора *S1* (называемого также **кросс-коннектором**).

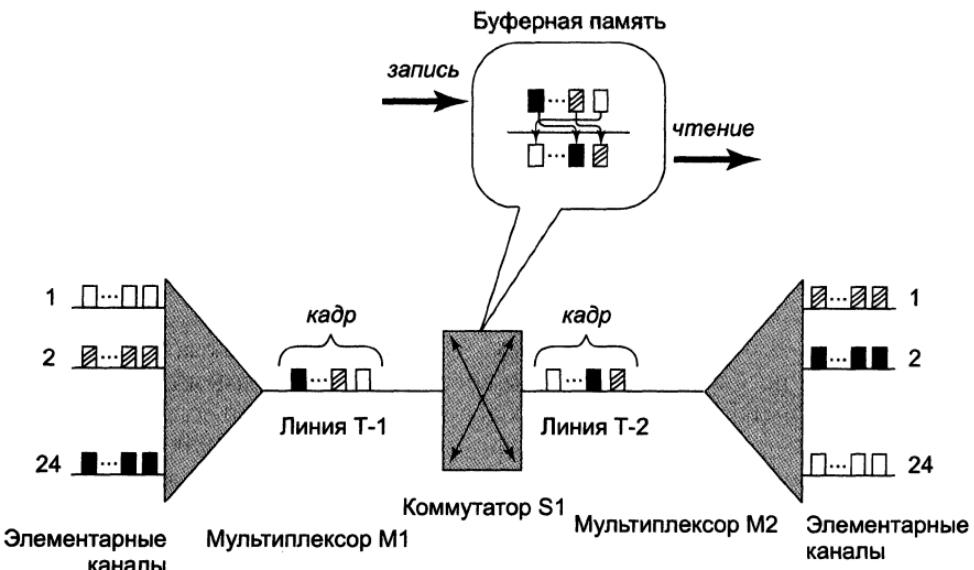


Рис. 4.2. Коммутация каналов в сети PDH

Аппаратура сетей TDM — мультиплексоры и коммутаторы — работает в режиме разделения времени, поочередно обслуживая в течение цикла своей

¹ Когда аббревиатура TDM указывается без уточнения режима работы, она всегда обозначает синхронный режим TDM.

² Мультиплексоры TDM выполняют как функции мультиплексирования, так и функции демультиплексирования и реализуются в виде одного устройства.

работы все абонентские каналы. Цикл равен 125 мкс, что соответствует периоду следования замеров голоса в цифровом абонентском канале. Это значит, что мультиплексор или коммутатор успевает вовремя обслужить любой абонентский канал и передать его очередной замер далее по сети. Каждому соединению выделяется один квант времени цикла работы аппаратуры, называемый также **тайм-слотом**. Длительность тайм-слота зависит от числа абонентских каналов, обслуживаемых мультиплексором или коммутатором.

В сети, показанной на рисунке, путем коммутации создано 24 канала, каждый из которых связывает пару абонентов. В частности, абонент, подключенный к входному каналу 1, связан с абонентом, подключенным к выходному каналу 24, абонент входного канала 2 связан с абонентом выходного канала 1, аналогично скоммутированы между собой абоненты входного канала 24 и выходного канала 2. Мультиплексор *M1* принимает информацию от абонентов по входным каналам, каждый из которых передает данные со скоростью 1 байт каждые 125 мкс (64 Кбит/с). В каждом цикле мультиплексор выполняет следующие действия:

1. Прием от каждого канала очередного байта данных.
2. Составление из принятых байтов кадра.
3. Передача кадра на выходной канал с битовой скоростью, равной 24×64 Кбит/с, что примерно составляет 1,5 Мбит/с.

Порядок следования байта в кадре соответствует номеру входного канала, от которого этот байт получен. Коммутатор *S1* принимает кадр по скоростному каналу от мультиплексора и записывает каждый байт из него в отдельную ячейку своей буферной памяти, причем в том порядке, в котором байты были упакованы в уплотненный кадр. Для выполнения операции коммутации байты извлекаются из буферной памяти не в порядке поступления, а в том порядке, который соответствует поддерживаемым в сети соединениям абонентов. В рассматриваемом примере коммутатор *S1* коммутирует входные каналы 1, 2 и 24 с выходными каналами 24, 2 и 1 соответственно. Для выполнения этой операции 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 байты кадра и распределяет их по своим нескольким выходным каналам, при этом он также считает, что порядковый номер байта в кадре соответствует номеру выходного канала.

Нарушение синхронности разрушает требуемую коммутацию абонентов, так как при этом изменяется относительное положение слота, а значит, теряется адресная информация. Поэтому оперативное перераспределение тайм-слотов между различными каналами в оборудовании TDM невозможно. Даже 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 будет передан пустым.

В общем случае сети TDM могут поддерживать либо режим динамической коммутации, либо режим постоянной коммутации, а иногда и оба эти режима. Цифровые телефонные сети поддерживают **динамическую коммутацию** по инициативе абонентов сети.

Основным режимом сетей PDH является **постоянная коммутация** (название *кросс-коннектор* как раз отражает постоянство соединений). Как правило, конфигурирование соединений PDH выполняется с помощью системы управления, в небольших сетях это делается вручную.

Иерархия скоростей

В плане соединения крупных телефонных станций каналы Т-1 представляли собой слишком слабые и негибкие средства мультиплексирования, поэтому была реализована идея образования каналов с *иерархией скоростей*. Четыре канала типа Т-1 объединили в канал следующего уровня цифровой иерархии — Т-2, передающий данные со скоростью 6,312 Мбит/с. Канал Т-3, образованный путем объединения семи каналов Т-2, имеет скорость 44,736 Мбит/с. Канал Т-4 объединяет 6 каналов Т-3, в результате его скорость равна 274 Мбит/с. Эта технология получила название **системы Т-каналов**.

Технология систем Т-каналов была стандартизована Американским национальным институтом стандартов (American National Standard Institute, ANSI) и получила название Plesiochronous Digital Hierarchy (PDH). В ходе стандартизации возникла несовместимость американской и международной версий стандарта PDH. Аналогом системы Т-каналов в международном стандарте являются каналы типа **E-1, E-2 и E-3** с отличающимися скоростями — соответственно 2,048, 8,448 и 34,368 Мбит/с. Несмотря на различия, в американской и международной версиях технологии цифровой иерархии принято использовать одни и те же обозначения для иерархии скоростей — DS_n (Digital Signal n). Скорость **DS0** соответствует скорости **элементарного канала 64 Кбит/с**.

Сети SONET/SDH

Практика выявила несколько недостатков технологии PDH, основными из которых являются следующие.

- Сложность и неэффективность операций мультиплексирования и демультиплексирования пользовательских данных, когда, например, для ответвления канала Е-1 из канала Е-3 последний необходимо сначала демультиплексировать на каналы Е-2 и только потом определенный канал Е-2 из этого набора демультиплексировать на каналы Е-1.

- Отсутствие в PDH средства обеспечения отказоустойчивости, то есть в этой технологии не поддерживаются процедуры автоматического переключения на резервный канал в случае выхода основного из строя.
- Недостаточная производительность даже на верхнем уровне иерархии скоростей.

Стандартизация

Указанные недостатки были учтены и преодолены разработчиками технологии **синхронных оптических сетей** (Synchronous Optical NET, SONET). Эта технология была стандартизована американским институтом ANSI. Позже был стандартизован международный вариант технологии SONET, который получил название **синхронной цифровой иерархии** (Synchronous Digital Hierarchy, SDH). Стандарт SONET был доработан так, чтобы аппаратура и сети SDH и SONET являлись совместимыми.

В стандарте SDH все уровни скоростей (и, соответственно, форматы кадров для этих уровней) имеют общее название **синхронный транспортный модуль уровня N** (Synchronous Transport Module level N, STM-N). Начальная скорость STM-1 равна 155 Мбит/с. Оборудование SDH поддерживает коэффициент кратности скоростей равный 4, то есть скорости STM-4, STM-16, STM-64 и STM-256 (40 Гбит/с).

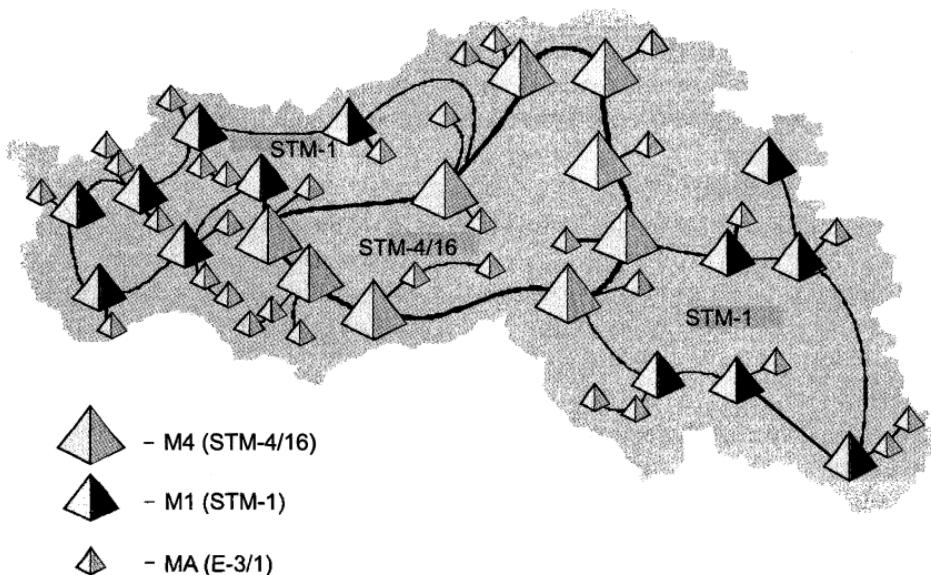


Рис. 4.3. Сеть SDH с сетями доступа PDH

Оборудование сетей SDH/SONET, так же как сетей PDH, состоит из мультиплексоров и кросс-коннекторов. Сети PDH обычно используются как сети доступа к сетям SDH. В примере на рис. 4.3 сеть SDH состоит

из магистральных кросс-коннекторов $M4$, работающих на скорости STM-4 и STM-16 (622 Мбит/с и 2,5 Гбит/с), мультиплексоров $M1$ технологии SDH, работающих на скорости STM-1 (155 Мбит/с). Сети доступа образованы мультиплексорами MA технологии PDH, работающими на скоростях E-3 (34 Мбит/с) и E-1 (2 Мбит/с).

Схема мультиплексирования SDH очень гибкая, она позволяет извлечь из потока данных высокого уровня иерархии скорости подпоток меньшей скорости без последовательного демультиплексирования высокоскоростного потока на его составляющие. Например, из потока STM-16 можно непосредственно ответвить поток STM-1. Техника мультиплексирования SDH основана на использовании так называемых **виртуальных контейнеров** (Virtual Container, VC) разного уровня скорости, которые инкапсулируются друг друга. Мультиплексоры SDH также называют **мультиплексорами ввода-выхода** (Add-Drop Multiplexers, ADM).

Синхронизация

Мультиплексоры SDH требуют для своей работы очень точной взаимной синхронизации (важность этого свойства видно из названия технологии). Такая синхронизация обеспечивается одним или несколькими внешними эталонными атомными часами, снабжающими своими синхроимпульсами магистральные мультиплексоры сети SDH. Мультиплексоры SDH более низких уровней иерархии синхронизируются другим способом — они извлекают синхросигналы из заголовков кадров, поступающих от магистральных мультиплексоров. В целом сеть синхронизации является важным элементом сети SDH, требующим особого внимания проектировщика сети.

Отказоустойчивость

Одной из сильных сторон первичных сетей SDH является разнообразный набор средств отказоустойчивости, который позволяет сети быстро (за десятки миллисекунд) восстановить работоспособность в случае отказа какого-либо ее элемента — линии связи, порта или карты мультиплексора, мультиплексора в целом.

В SDH в качестве общего названия механизмов отказоустойчивости используется термин **автоматическое защитное переключение** (Automatic Protection Switching, APS), отражающий факт перехода (переключения) на резервный путь или резервный элемент мультиплексора при отказе основного. Существует несколько типов защиты APS, из которых наиболее популярны защита сетевого соединения и защита на основе разделения кольца.

Задача сетевого соединения основана на установлении в сети произвольной топологии двух соединений между конечными точками: рабочего и резервного. Трафик передается параллельно по этим двум соединениям, и точка приема выбирает, трафик от какого соединения она считает основным. Выбор осуществляется на основании информации о качестве сигнала, передаваемой

в кадрах SDH. В случае отказа рабочего соединения точка приема переходит на прием информации от резервного канала, при этом переход осуществляется очень быстро, обычно в пределах 50 мс. Недостатком этого способа является его избыточность, так как фактически вместо одного канала в сети работает два.

Задита на основе разделения кольца более экономична, она не создает в сети параллельных каналов, а пытается направить информацию в обратном направлении, если какая-либо секция кольца отказывает. Естественно, этот способ работает только в кольцевых топологиях SDH. Нужно отметить, что кольцевые топологии SDH очень популярны из-за своей экономичности в плане реализации защиты соединений.

Техника корректировки ошибок (Forward Error Control, FEC) обычно используется мультиплексорами SDH на скоростях 2,5 Гбит/с и выше. Эта техника основана на применении самокорректирующих кодов, позволяющих исправлять искажения битов данных «на лету», то есть не прибегая к их повторной передаче, а используя избыточную часть кода. Такая техника может существенно повысить эффективную скорость передачи данных при наличии помех или сбоев в работе приемопередатчиков.

Сети DWDM

Технология **уплотненного волнового мультиплексирования** (Dense Wave Division Multiplexing, DWDM) предназначена для создания оптических магистралей нового поколения, работающих на мультигигабитных и терабитных скоростях.

Такой революционный скачок производительности обеспечивается принципиально иным, нежели в SDH, методом мультиплексирования — информация в оптическом волокне передается одновременно большим количеством световых волн — лямбда. Термин **лямбда** возник в связи с традиционным для физики обозначением длины волны λ . Сети DWDM работают по принципу коммутации каналов, при этом каждая световая волна представляет собой отдельный *спектральный канал* и переносит отдельный поток данных.

На рис. 4.4 показан пример мультиплексирования двух волн λ_1 и λ_2 мультиплексорами DWDM. Каждый из мультиплексоров DWDM принимает так называемые «неокрашенные» сигналы (в нашем примере — от мультиплексоров SDH, но это может быть любое оборудование, например, IP-маршрутизаторы), то есть оптические сигналы одной из принятых в оптических сетях длин волн: 850, 1300 или 1550 нм (как вы помните, они соответствуют центрам окон прозрачности оптического волокна). Затем мультиплексоры DWDM преобразуют принятые сигналы в волны определенной

длины для каждого их интерфейсов, в нашем примере – λ_1 и λ_2 , и в этой форме они передаются по одному и тому же волокну, не интерферируя и не искажая информацию, передаваемую каждой волной. Принимающий мультиплексор DWDM выполняет демультиплексирование волн из общего сигнала, преобразуя каждую волну в «неокрашенную», которую понимают обычные интерфейсы мультиплексоров SDH.

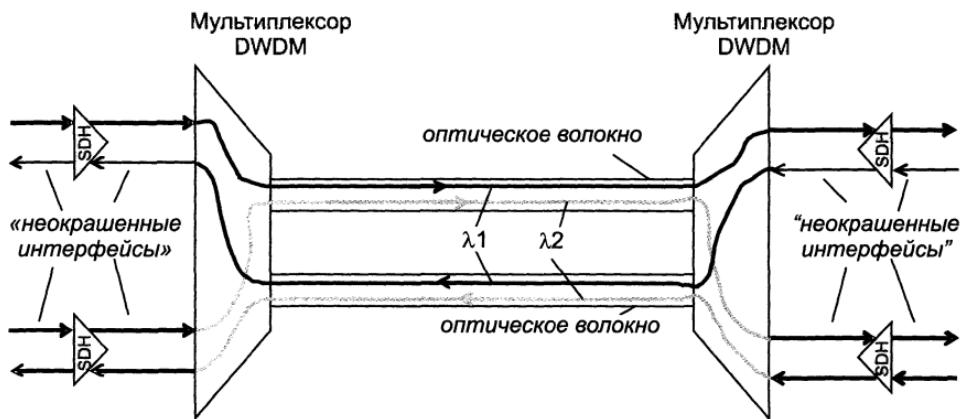


Рис. 4.4. Принцип мультиплексирования волн различной длины в одном волокне

Оборудование DWDM не занимается непосредственно проблемами передачи данных на каждой волне, то есть способом кодирования информации и протоколом ее передачи. Его основными функциями являются операции **мультиплексирования** и **демультиплексирования**, а именно – объединение различных волн в одном световом пучке и выделение информации каждого спектрального канала из общего сигнала. Наиболее развитые устройства DWDM могут также **коммутировать** волны.

Оборудование DWDM позволяет передавать по одному оптическому волокну 32 и более волн разной длины в окне прозрачности 1550 нм, при этом каждая волна может переносить информацию со скоростью до 10 Гбит/с (при применении протоколов технологий STM или 10 Gigabit Ethernet для передачи информации на каждой волне). В настоящее время ведутся работы по повышению скорости передачи информации на одной длине волны до 40–80 Гбит/с.

У технологии DWDM имеется предшественница – **технология волнового мультиплексирования** (Wave Division Multiplexing, WDM), в которой используются существенно большее расстояние между длинами волн, чем в DWDM, из-за чего мультиплексирование в последней и назвали «уплотненным».

В оборудовании DWDM первых поколений в качестве интерфейсов доступа обычно использовались интерфейсы стандарта SDH. Это приводило к тому,

что, несмотря на принципиальную возможность передавать дискретные данные любой технологии, оборудование DWDM этих поколений передавало данные кадрами SDH, а если было необходимо передать данные, например, Gigabit Ethernet, то их нужно было предварительно упаковать в кадры SDH, а потом уже передавать по сети DWDM. При таком подходе пропускная способность волновых каналов DWDM расходуется не очень эффективно, к тому же стоимость интерфейсов SDH очень высока.

Сети OTN

Для решения этой проблемы была разработана новая технология **оптических транспортных сетей** (Optical Transport Networks, OTN), ориентированная на магистральные сети, так как она поддерживает только верхний диапазон скоростей, оставляя мультиплексирование низкоскоростных потоков технологии SDH (или Ethernet).

В сетях OTN поддерживается три формата кадра, соответствующих следующей иерархии скоростей: OTU1 (2,7 Гбит/с), OTU2 (10,7 Гбит/с) и OTU3 (43 Гбит/с). Как следует из этого перечня, OTN обеспечивает мультиплексирование с коэффициентом 4, то есть каждый кадр более высокого уровня иерархии состоит из 4 кадров более низкого уровня.

Форматы кадров OTN позволяют размещать в своем поле данных кадры практически любой современной технологии передачи данных: SDH, Ethernet, Fibre Channel. В этом состоит одно из отличий технологии OTN от SDH, изначально ориентированной только на перенос трафика телефонных сетей и поэтому имеющей линейку скоростей, кратную скорости 64 Кбит/с. Другим достоинством OTN является относительная простота схем мультиплексирования, которая объясняется наличием всего трех уровней в ее линейке скоростей.

В OTN, так же как и в SDH, выполняется процедура корректировки ошибок FEC, но самокорректирующийся код, используемый в OTN, превосходит по эффективности код, применяемый в SDH.

Технология Frame Relay

История стандарта

Пакетная технология глобальных сетей **Frame Relay** появилась в конце 80-х годов в связи с распространением высокоскоростных и надежных цифровых каналов технологий PDH и SDH. До этого основной технологией глобальных сетей являлась технология X.25, сложный стек которой был рассчитан на низкоскоростные аналоговые каналы, отличавшиеся к тому же высоким уровнем помех, и, следовательно, ошибок в передаче данных. Особенностью Frame Relay является простота; освободившись от многих ненужных в современном телекоммуникационном мире функций, эта технология

предоставляет только тот минимум услуг, который необходим для доставки кадров адресату. Вместе с тем разработчики технологии Frame Relay сделали важный шаг вперед, предоставив пользователям сети *гарантию пропускной способности* сетевых соединений — свойство, которое до появления Frame Relay технологии пакетных сетей стандартным способом не поддерживали.

Техника продвижения кадров

Технология Frame Relay основана на использовании техники *виртуальных каналов*, которую мы кратко рассмотрели в главе 1. Техника виртуальных каналов является компромиссом между неопределенностью дейтаграммного способа продвижения пакетов, используемого, например, в сетях Ethernet и IP, и жесткостью техники коммутации каналов, которая свойственна технологиям первичных и телефонных сетей.

Операторы глобальных сетей достаточно долго, вплоть до революционного распространения Интернета в середине 90-х, отдавали предпочтение технике виртуальных каналов. По сравнению с дейтаграммным методом продвижения данных техника виртуальных каналов предоставляет более широкие возможности по контролю над соединениями между узлами и путями прохождения информационных потоков через сеть. Виртуальные каналы обладают также некоторыми преимуществами и по отношению к технике коммутации каналов, позволяя более гибко распоряжаться пропускной способностью. Операторы в этих условиях могут вмешиваться в распределение ресурсов между пользователями так, чтобы каждый пользователь получал именно те услуги, на которые он подписался.

Рассмотрим технику виртуальных каналов сетей Frame Relay на примере сети, изображенной на рис. 4.5.

Для того чтобы конечные узлы сети — компьютеры *C1*, *C2*, *C3* и сервер *C4* — могли обмениваться данными, в сети необходимо предварительно проложить виртуальные каналы. В нашем примере установлено три таких канала — между компьютерами *C1* и *C2* через коммутатор *S1*; между компьютером *C1* и сервером *C4* через коммутаторы *S1* и *S2*; между компьютером *C3* и сервером *C4* через коммутатор *S2*.

Виртуальные каналы Frame Relay могут быть как **однонаправленными** (то есть способными передавать кадры только в одном направлении), так и **дву направленными**.

Будем считать, что в примере на рис. 4.5 были установлены двунаправленные каналы.

Процедура установления виртуальных каналов Frame Relay заключается в формировании таблиц коммутации в коммутаторах сети. Такие процедуры могут выполняться как вручную, так и системами управления сетью.

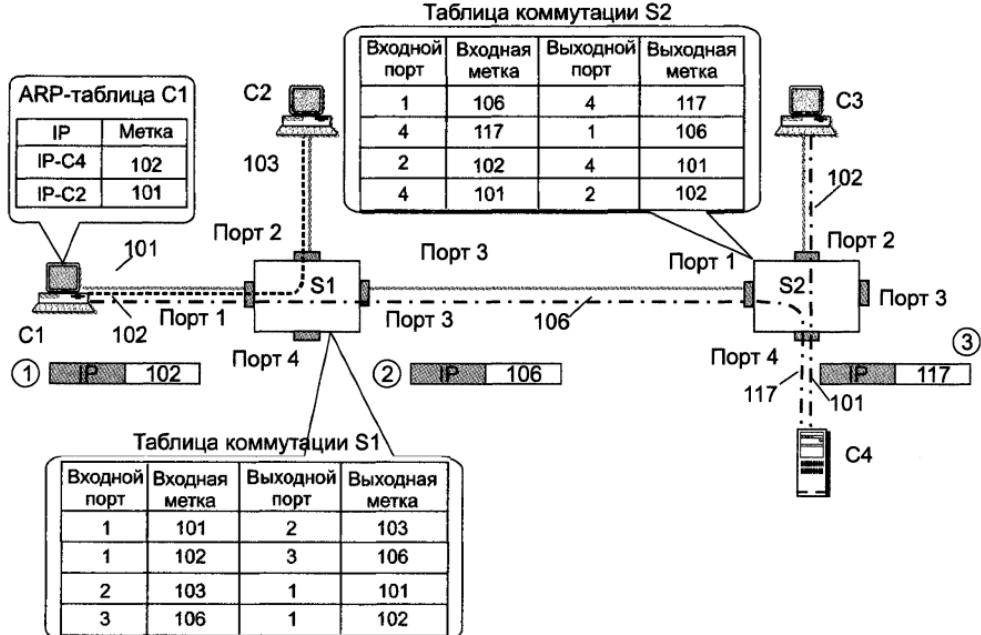


Рис. 4.5. Продвижение кадров вдоль виртуальных каналов

Виртуальные каналы Frame Relay относятся к типу **постоянных виртуальных каналов** (Permanent Virtual Circuits, PVC), они заранее устанавливаются по командам оператора сети.

В таблице коммутации каждого коммутатора должны быть сделаны две записи (для каждого из двух направлений) о каждом из виртуальных каналов, проходящих через данный коммутатор.

Запись таблицы коммутации состоит из четырех основных полей:

- номер входного порта канала;
- входная метка канала в поступающих на входной порт пакетах;
- номер выходного порта;
- выходная метка канала в передаваемых через выходной порт пакетах.

Например, вторая запись в таблице коммутации коммутатора S1 (запись 1-102-3-106) означает, что все пакеты, которые поступят на порт 1 с идентификатором виртуального канала 102, будут продвигаться на порт 3, а в поле идентификатора виртуального канала появится новое значение – 106. Так как виртуальные каналы в нашем примере двунаправленные, то для каждого канала в таблице коммутации должно существовать две записи, описывающие преобразование метки в каждом из направлений. Так, для записи 1-102-3-106 существует запись 3-106-1-102.

- Метки виртуального канала имеют локальное для коммутатора и его порта значение, то есть они никаким образом не принимаются во внимание на портах других коммутаторов.
- Комбинации «метка-порт» должны быть уникальными в пределах одного коммутатора.
- Кроме того, непосредственно соединенные порты двух коммутаторов должны использовать согласованные значения меток для каждого виртуального канала, проходящего через эти порты.

После того как виртуальные каналы установлены, конечные узлы могут использовать их для обмена информацией. Для этого администратор сети должен для каждого конечного узла создать статические записи таблицы ARP. В каждой такой записи устанавливается соответствие между IP-адресом узла назначения и начальным значением метки виртуального канала, ведущего к этому узлу. Например, в таблице ARP компьютера *C1* должна присутствовать запись, отображающая IP-адрес сервера *C4* на метку 102 для виртуального канала, ведущего к серверу *C4*.

Давайте сейчас проследим путь одного кадра, отправленного компьютером *C1* серверу *C4*. При отправлении кадра (этап 1 на рис. 4.5) компьютер помещает в поле адреса начальное значение метки 102, взятое из его таблицы ARP.

Коммутатор *S1*, получив на порт 1 кадр с меткой 102, просматривает свою таблицу коммутации и находит, что такой кадр должен быть переправлен на порт 3, а значение метки в нем должно быть заменено на 106.

ПРИМЕЧАНИЕ

Операция по замене метки (label swapping) характерна для всех технологий, использующих технику виртуальных каналов. Может возникнуть законный вопрос: «А зачем менять значение метки на каждом коммутаторе? Почему бы не назначить каждому виртуальному каналу одно неизменяемое значение метки, которое играло бы роль физического адреса узла назначения?» Ответ состоит в том, что в первом случае уникальность меток достаточно обеспечивать в пределах каждого отдельного порта, а во втором — в пределах всей сети, что гораздо сложнее, так как требует наличия в сети централизованной службы назначения меток.

В результате действий коммутатора *S1* кадр отправляется через порт 3 к коммутатору *S2* (этап 2). Коммутатор *S2*, используя свою таблицу коммутации, находит соответствующую запись, заменяет значение метки на 117 и отправляет кадр узлу назначения — серверу *C4*. На этом обмен заканчивается, а при отправке ответа сервер *C4* задействует метку 117 как адрес виртуального канала, ведущего к компьютеру *C1*.

Как видно из этого описания, коммутация выполняется очень экономично, так как преобразования передаваемых кадров минимальны — они сводятся только к замене значения метки. В кадрах указывается только адрес

назначения, роль которого в сетях Frame Relay играет метка. В качестве адреса отправителя может быть использовано последнее значение метки — оно однозначно определяет путь в обратном направлении по виртуальному каналу, соединяющему получателя и отправителя.

Гарантии пропускной способности

Сети Frame Relay создавались для оказания коммерческих услуг операторов связи по передаче компьютерного трафика. Одним из новых и очень привлекательных для клиентов свойств услуг Frame Relay стала поддержка гарантий пропускной способности виртуальных соединений. Для каждого виртуального соединения в технологии Frame Relay определяется несколько параметров, связанных со скоростью передачи данных.

- ❑ **Согласованная скорость передачи данных** (Committed Information Rate, CIR) — гарантированная пропускная способность соединения; фактически сеть гарантирует передачу данных пользователя со скоростью предложенной нагрузки, если эта скорость не превосходит CIR.
- ❑ **Согласованная величина пульсации** (Committed Burst Size, Bc) — максимальное количество байтов, которое сеть будет передавать от данного пользователя за интервал времени T , называемый временем пульсации, соблюдая согласованную скорость CIR.
- ❑ **Дополнительная величина пульсации** (Excess Burst Size, Be) — максимальное количество байтов, которое сеть будет пытаться передать сверх установленного значения Bc за интервал времени T .

Второй параметр пульсации Be позволяет оператору сети дифференцированно обрабатывать кадры, которые не укладываются в профиль CIR. Обычно кадры, которые приводят к превышению пульсации Bc, но не превышают пульсации Bc + Be, сетью не отбрасываются, а обслуживаются, но без гарантий по скорости CIR. Для запоминания факта нарушения в кадрах Frame Realy имеется специальное поле возможности удаления (Discard Eligibility, DE). И только если превышен порог Bc + Be, кадры отбрасываются.

Если приведенные величины определены, то время T определяется следующей формулой:

$$T = Bc/CIR.$$

Можно рассматривать значения CIR и T в качестве варьируемых параметров, тогда производной величиной станет пульсация Bc. Обычно для контроля пульсаций трафика выбирается время T , равное 1–2 секундам при передаче компьютерных данных, и в диапазоне десятков–сотен миллисекунд при передаче голоса.

Соотношение между параметрами CIR, Bc, Be и T иллюстрирует рис. 4.6 (R — скорость в канале доступа; f_1-f_5 — кадры).

Работа сети описывается двумя линейными функциями, показывающими зависимость количества переданных битов от времени: $B = R \times t$ и $B = CIR \times t$.

Средняя скорость поступления данных в сеть составила на этом интервале R бит/с, и она оказалась выше CIR. На рисунке показан случай, когда за интервал времени T в сеть по виртуальному каналу поступило 5 кадров. Кадры f_1, f_2 и f_3 доставили в сеть данные, суммарный объем которых не превысил порог Bc , поэтому эти кадры ушли дальше транзитом с признаком $DE = 0$. Данные кадра f_4 , прибавленные к данным кадров f_1, f_2 и f_3 , уже превысили порог Bc , но еще не достигли порога $Bc + Be$, поэтому кадр f_4 также ушел дальше, но уже с признаком $DE = 1$. Данные кадра f_5 , прибавленные к данным предыдущих кадров, превысили порог $Bc + Be$, поэтому этот кадр был удален из сети.

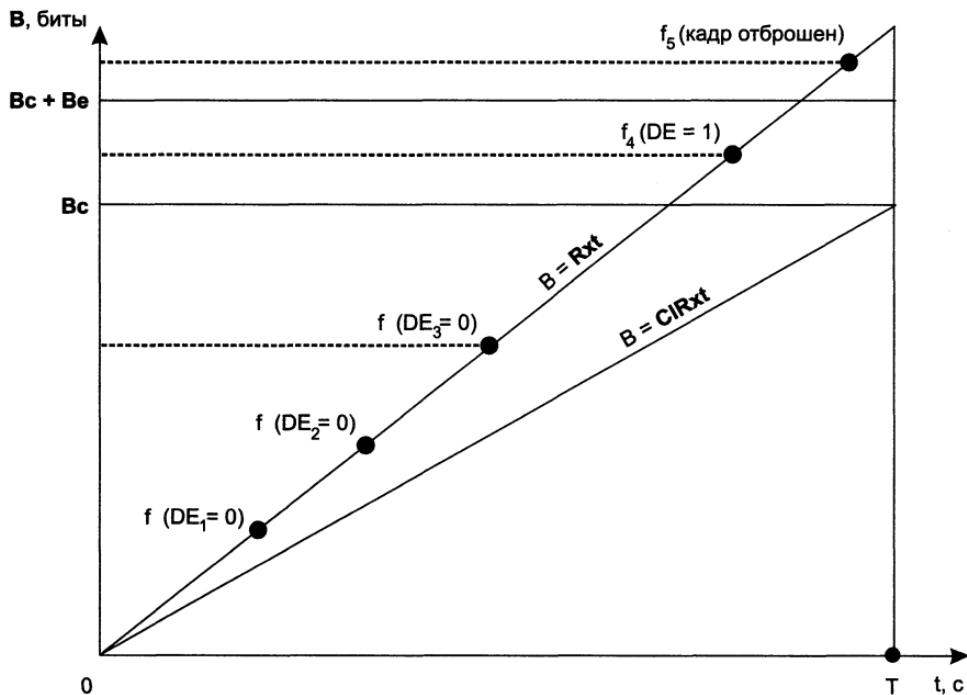


Рис. 4.6. Реакция сети на поведение пользователя

На рис. 4.7 приведен пример сети Frame Relay с пятью удаленными региональными отделениями корпорации. Обычно доступ к сети осуществляется по каналам с пропускной способностью, большей чем CIR. Но при этом пользователь платит не за пропускную способность канала, а за заказанные величины CIR, Bc и Be . Так, при применении в качестве линии доступа канала T-1 и заказа обслуживания со скоростью CIR, равной 128 Кбит/с, пользователь будет платить только за скорость 128 Кбит/с, а скорость канала T-1 в 1,5 Мбит/с окажет влияние на верхнюю границу возможной пульсации $Bc + Be$.

Параметры качества обслуживания могут быть разными для разных направлений виртуального канала. Так, на рисунке абонент 1 соединен

с абонентом 2 виртуальным каналом с меткой 136. При направлении от абонента 1 к абоненту 2 канал имеет среднюю скорость 128 Кбит/с с пульсациями $B_c = 256$ Кбит (интервал T составил 1 с) и $B_e = 64$ Кбит. А при передаче кадров в обратном направлении средняя скорость уже может достигать значения 256 Кбит/с с пульсациями $B_c = 512$ Кбит и $B_e = 128$ Кбит.

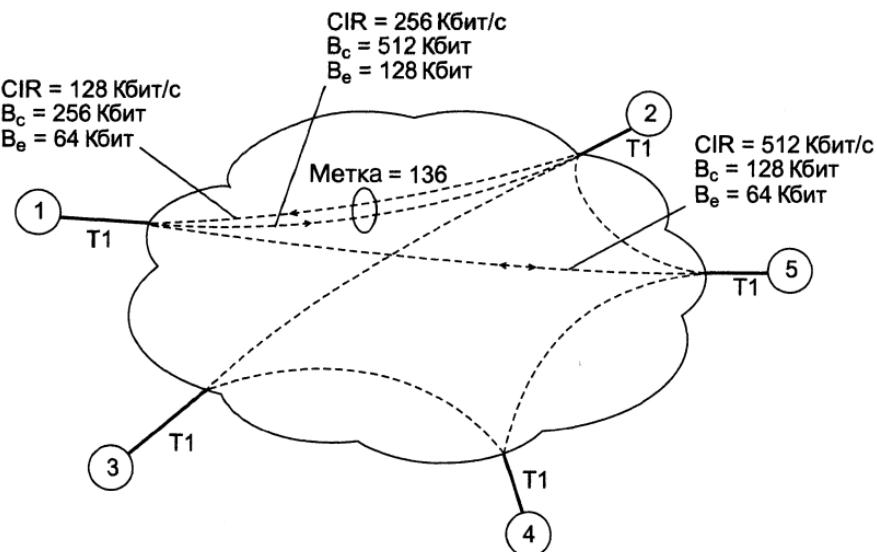


Рис. 4.7. Пример обслуживания в сети Frame Relay

Технология Frame Relay получила большое распространение в сетях операторов связи в 90-е годы из-за сочетания простоты и возможности гарантировать клиентам пропускную способность соединений. Тем не менее в последнее время популярность услуг Frame Relay резко упала, в основном это произошло из-за появления технологии MPLS, которая, так же как и Frame Relay, основана на технике виртуальных каналов и может гарантировать пропускную способность пользовательских соединений. Решающим преимуществом MPLS является ее тесная интеграция с технологией IP, в результате чего провайдерам легче формировать новые комбинированные услуги. Кроме того, функциональность MPLS поддерживается сегодня практически всеми маршрутизаторами среднего и высшего класса, так что применение MPLS не требует установки в сети отдельных коммутаторов. Мы рассмотрим эту технологию немного позже.

Технология ATM

Асинхронный режим передачи (Asynchronous Transfer Mode, ATM) — это технология, основанная на установлении виртуальных каналов и предназначенная для использования в качестве единого универсального транспорта для нового поколения сетей с интегрированным обслуживанием.

Под **интегрированным обслуживанием** здесь понимается способность сети передавать трафик разного типа: *чувствительный к задержкам* (например, голосовой) трафик и *эластичный*, то есть допускающий задержки в широких пределах трафик (например, трафик электронной почты или просмотра веб-страниц). Этим технология ATM принципиально отличается от технологии Frame Relay, которая изначально предназначалась только для передачи эластичного компьютерного трафика.

Кроме того, в цели разработчиков технологии ATM входило обеспечение широкой иерархии скоростей и возможности использования первичных сетей SDH для соединения коммутаторов ATM. В результате производители оборудования ATM ограничились первыми двумя уровнями иерархии скоростей SDH, то есть STM-1 (155 Мбит/с) и STM-4 (622 Мбит/с).

Ячейки ATM

В технологии ATM для переноса данных используются **ячейки**. Принципиально ячейка отличается от кадра только тем, что имеет, во-первых, *фиксированный*, во-вторых, *небольшой* размер. Длина ячейки составляет 53 байта, а поля данных — 48 байт. Именно такие размеры позволяют сети ATM передавать чувствительный к задержкам голосовой и видеотрафик с необходимым уровнем качества.

Мы уже упоминали в главе 1, что задержки передачи пакетов в сетях с коммутацией пакетов (к которым относится и ATM) являются следствием самого принципа коммутации, так как каждый пакет передается индивидуально и, возможно, с длительной буферизацией из-за наличия очередей в каждом коммутаторе или маршрутизаторе сети. Для голосового и видеотрафика задержки передачи отдельных пакетов приводят к весьма плохим последствиям, так как страдает качество воспроизведенного на приемной стороне голоса или изображения. Те, кто пользуется Интернет-телефонией, знакомы с подобными негативными эффектами, например,искажением тембра голоса собеседника (вплоть до потери распознаваемости как самой личности, так и слов, которые она произносит), появлением эха и т. п.

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. 4.8).

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.



Рис. 4.8. Дискретная модуляция непрерывного процесса

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: $8000 \times 8 = 64\,000$ бит/с или 64 Кбит/с. Такой голосовой канал называют **элементарным каналом цифровых телефонных сетей**.

Передача непрерывного сигнала в дискретном виде требует от сетей жесткого соблюдения временного интервала в **125 мкс** между соседними замерами, в противном случае исходный сигнал восстанавливается неверно, что приводит к искажению голоса, изображения или другой мультимедийной информации, передаваемой по цифровым сетям. Так, сдвиги между замерами в 200 мс приводят к невозможности распознавания произносимых слов.

В то же время потеря одного замера при соблюдении синхронности между остальными замерами практически не сказывается на воспроизведимом звуке. Это происходит за счет сглаживающих устройств в цифроаналоговых преобразователях, работа которых основана на свойстве инерционности любого физического сигнала — амплитуда звуковых колебаний не может мгновенно измениться на большую величину.

В целом параметры, характеризующие влияние очередей в сети на точность доставки пакетов во времени, называются **параметрами качества обслуживания** (Quality of Service, QoS).

К этим параметрам обычно относят:

- ❑ задержку доставки пакета (желательно не более 150 мс для голоса);
- ❑ вариацию задержки доставки пакета (желательно не более 80–100 мс для голоса);
- ❑ долю потерь пакетов в очередях (желательно менее 1 %).

Одним из методов обеспечения требуемых параметров QoS является *приоритетное обслуживание* пакетов (кадров, ячеек), переносящих чувствительный к задержкам трафик. Приоритетная организация очередей поддерживается сегодня как IP-маршрутизаторами, так и коммутаторами Ethernet.

Одной из причин, по которым разработчики выбрали небольшой и фиксированный размер ячейки ATM, является стремление снизить **задержки ожидания в очереди** для приоритетных ячеек. Дело в том, что при передаче данных в сетях реализуется алгоритм обслуживания с *относительными приоритетами*, в соответствии с которым поступившая в узел ячейка с наивысшим приоритетом, хотя и направляется во главу очереди, не прерывает обработку текущей ячейки, а ждет завершения ее передачи. Таким образом, наличие длинных низкоприоритетных ячеек может привести к слишком большому времени ожидания в очереди даже для ячеек с наивысшим приоритетом.

Другой причиной явилось стремление ограничить еще одну составляющую задержки доставки данных – задержку пакетизации. **Задержка пакетизации** равна времени, в течение которого первый замер голоса ждет момента окончательного формирования пакета и отправки его по сети.

Механизм образования этой задержки иллюстрирует рис. 4.9.

На рисунке показан голосовой кодек – устройство, которое представляет голос в цифровой форме. Пусть он выполняет замеры голоса в соответствии со стандартной частотой 8 кГц (то есть через каждые 125 мкс), кодируя каждый замер одним байтом данных. Если мы используем для передачи голоса кадры Ethernet максимального размера, то в один кадр помещается 1500 замеров голоса. В результате первый замер, помещенный в кадр Ethernet, вынужден ждать отправки кадра в сеть $(1500 - 1) \times 125 = 187\,375$ мкс, или около 187 мс. Это весьма большая задержка для голосового трафика. Рекомендации стандартов говорят о величине 150 мс как о максимально допустимой *суммарной* задержке голоса, в которую задержка пакетизации входит как одно из слагаемых.

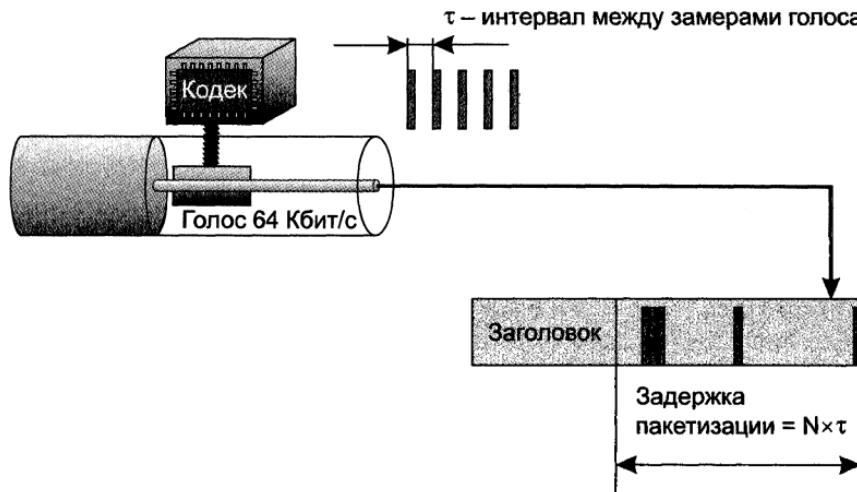


Рис. 4.9. Задержка пакетизации

ВНИМАНИЕ

Важно отметить, что задержка пакетизации не зависит от битовой скорости протокола, она зависит только от частоты работы кодека и размера поля данных кадра. Это отличает ее от задержки ожидания в очереди, которая снижается с возрастанием битовой скорости.

Размер ячейки ATM в 53 байта с полем данных 48 байт стал результатом компромисса между требованиями, предъявляемыми к сети при передаче эластичного и чувствительного к задержкам трафиков. Можно сказать также, что компромисс был достигнут между телефонистами и компьютерщиками — первые настаивали на размере поля данных в 32 байта, а вторые — в 64 байта.

При размере поля данных в 48 байт одна ячейка ATM обычно переносит 48 замеров голоса, которые делаются с интервалом в 125 мкс. Поэтому первый замер должен ждать примерно 6 мс, прежде чем ячейка будет отправлена по сети. Именно по этой причине телефонисты боролись за уменьшения размера ячейки, так как 6 мс — это задержка, близкая к пределу, за которым начинаются нарушения качества передачи голоса. При выборе размера ячейки в 32 байта задержка пакетизации составила бы 4 мс, что гарантировало бы более качественную передачу голоса. А стремление компьютерных специалистов увеличить поле данных хотя бы до 64 байт вполне понятно — при этом повышается полезная скорость передачи данных. Избыточность служебных данных при использовании 48-байтного поля данных составляет 10 %, а при использовании 32-байтного поля данных она сразу повышается до 16 %.

Виртуальные каналы ATM

В сетях ATM поддерживается два типа виртуальных каналов:

- **постоянный виртуальный канал** (Permanent Virtual Circuits, **PVC**);
- **коммутируемый виртуальный канал** (Switched Virtual Circuits, **SVC**), создание которого происходит динамически по инициативе конечного узла с использованием автоматической процедуры.

Каналы PVC аналогичны каналам такого же типа в сетях Frame Relay, а для поддержки динамически устанавливаемых каналов SVC в технологии ATM добавлен специальный протокол **сигнализации** — это протокол, с помощью которого абоненты сети могут оперативно устанавливать каналы SVC. Такой тип протокола используется в телефонных сетях для установления соединения между телефонами абонентов. Для того чтобы протокол сигнализации мог работать, конечные узлы сети ATM получили глобально уникальные 20-разрядные адреса, иначе абонент — инициатор установления виртуального канала не смог бы указать, с каким абонентом он хочет связаться.

С целью обеспечения масштабируемости в сетях ATM введено два уровня иерархии виртуальных каналов: **виртуальный путь** (virtual path) и **виртуальное соединение** (virtual circuit). Виртуальный путь определяется старшей частью номера метки виртуального канала, а виртуальное соединение — младшей. Каждый виртуальный путь включает в себя до 4096 виртуальных соединений, проходящих внутри этого пути. Достаточно определить маршрут для пути, и все соединения, которые находятся внутри этого пути, будут ему следовать.

Категории услуг ATM

Для поддержания требуемого качества обслуживания и рационального расходования ресурсов в технологии ATM реализовано несколько служб. Услуги этих служб разбиты на категории, которые, в общем, соответствуют классам трафика, поступающим на вход сети.

Всего на уровне протокола ATM определено пять категорий услуг:

- **CBR (Constant Bit Rate)** — для трафика с постоянной битовой скоростью, например, голосового;
- **rtVBR (real-time Variable Bit Rate)** — для трафика с переменной битовой скоростью, требующего соблюдения средней скорости передачи данных и синхронизации источника и приемника (примером является видео трафик с переменной битовой скоростью, который вырабатывают многие видеокодеки из-за использования опорных кадров и кадров, описывающих изменения изображения относительно опорного кадра);

- **nrtVBR** (non real-time Variable Bit Rate) – для трафика с переменной битовой скоростью, требующего соблюдения средней скорости передачи данных и не требующего синхронизации источника и приемника;
- **ABR** (Available Bit Rate) – для трафика с переменной битовой скоростью, требующего соблюдения некоторой минимальной скорости передачи данных и не требующего синхронизации источника и приемника;
- **UBR** (Unspecified Bit Rate) – для трафика, не предъявляющего требований к скорости передачи данных и синхронизации источника и приемника.

Отсюда видно, что сети ATM отличаются от сетей Frame Relay большей степенью соответствия услуги требованиям трафика определенного типа, так как в сетях ATM нужный уровень обслуживания задается не только численными значениями параметров CIR, Bs и Be, но и самой категорией услуги.

Технология ATM, как и технология Frame Relay, пережила пик своей популярности, и сейчас области ее применения быстро сокращаются. Одной из причин этого стало появление сетей DWDM и расширение верхней границы технологии Ethernet, предоставляющих относительно дешевую пропускную способность. Еще одной причиной снижения интереса к ATM стала сложность этой технологии. В частности, некоторые проблемы возникают из-за использования ячеек маленького размера – на высоких скоростях оборудование с трудом справляется с обработкой таких интенсивных потоков ячеек (сравните количество кадров Ethernet максимальной длины с количеством ячеек ATM, необходимых для передачи одного и того же объема информации с той же самой скоростью).

Как и в случае с Frame Relay, появление технологии MPLS, которая, с одной стороны, обладает некоторыми свойствами ATM, например, поддерживает детерминированность маршрутов (это общее свойство технологий, основанных на технике виртуальных путей), а с другой стороны, использует кадры любого формата и тесно интегрирована с IP, усугубило положение ATM. Одной из областей, где ATM по-прежнему удерживает позиции, является широкополосный доступ в Интернет. Если вы посмотрите на конфигурацию вашего домашнего маршрутизатора ADSL, то, скорее всего, увидите там записи, относящиеся к стеку ATM.

Технология MPLS

Технология многопротокольной коммутации с помощью меток (Multi-Protocol Label Switching, MPLS) считается сегодня многими специалистами одной из самых перспективных транспортных технологий. Эта технология объединяет преимущества техники виртуальных каналов с функциональностью стека TCP/IP.

Объединение происходит за счет того, что одно и то же сетевое устройство, называемое **коммутирующим по меткам маршрутизатором** (Label Switch Router, LSR), выполняет функции как IP-маршрутизатора, так и коммутатора виртуальных каналов. Причем это не механическое объединение двух устройств, а тесная интеграция, когда функции каждого устройства дополняют друг друга и используются совместно.

Впервые идея объединения маршрутизации и коммутации в одном устройстве была реализована в середине 90-х годов, когда на рынке появились комбинированные устройства IP/ATM. В этих устройствах была реализована новая технология **IP-коммутации** (IP switching), которая решала проблему ускорения продвижения IP-пакетов через сеть совместными усилиями протоколов стеков IP и ATM. В то время протокол IP часто работал поверх слоя ATM, при этом каналы ATM использовались для быстрой передачи данных (часто с поддержкой гарантированной пропускной способности и QoS) между пограничными IP-маршрутизаторами сети провайдера, осуществляя тем самым «обход», или «прокол», промежуточных магистральных IP-маршрутизаторов. IP-маршрутизаторы того времени были существенно медленнее коммутаторов ATM, поэтому эффект от такого «прокола» был существенным. Однако у данного способа был недостаток — он плохо работал для кратковременных потоков, так как время установления динамического соединения ATM для них было соизмеримо со временем передачи данных этого потока или превышало его.

Объединение функций IP и ATM в одном устройстве позволило решить эту проблему и к тому же устраниТЬ дублирование протоколов маршрутизации, так как топология сети для обоих стеков (то есть IP и ATM) была одинаковой.

Технология IP-коммутации была сразу замечена операторами связи и стала достаточно популярной. На основе нескольких вариантов фирменных технологий рабочая группа IETF, состоящая из специалистов различных компаний, создала в конце 90-х годов технологию MPLS.

LSR и таблица продвижения данных

Протоколы маршрутизации используются для определения топологии сети, а для продвижения данных внутри границ сети одного провайдера применяется техника виртуальных каналов.

Этот главный принцип технологий-предшественниц, таких как IP-коммутация, в MPLS был сохранен.

Принцип объединения протоколов различных технологий иллюстрируют рис. 4.10 и 4.11. На первом из них показана упрощенная архитектура

стандартного IP-маршрутизатора, на втором — архитектура комбинированного устройства LSR, поддерживающего технологию MPLS.

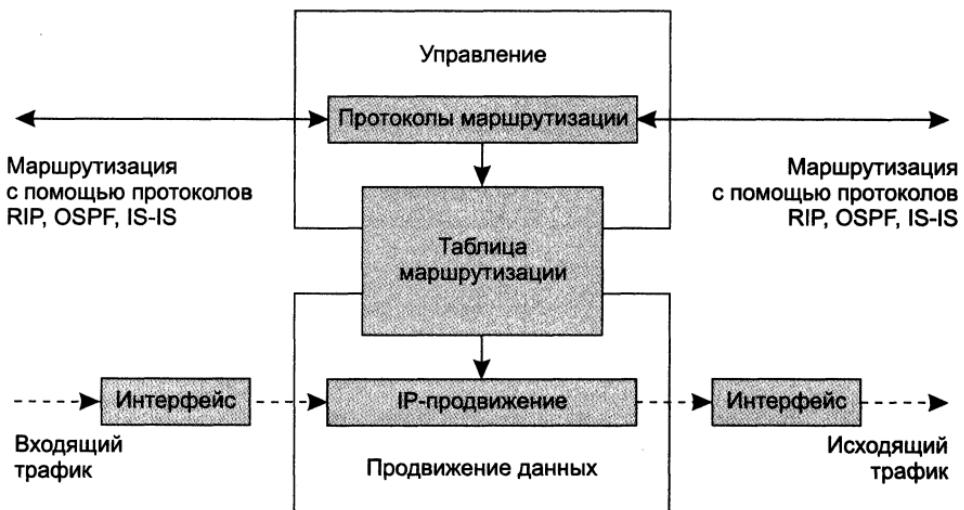


Рис. 4.10. Архитектура IP-маршрутизатора

Так как устройство LSR выполняет все функции IP-маршрутизатора, оно содержит все блоки последнего, а для поддержки функций MPLS в LSR включен ряд дополнительных блоков, относящихся как к управлению, так и к продвижению данных.

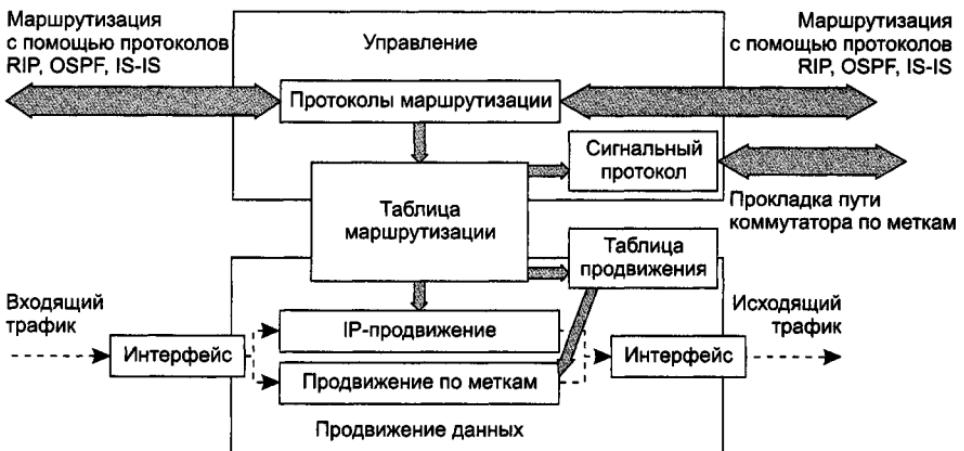


Рис. 4.11. Архитектура LSR

В качестве примера можно указать на блок *продвижения по меткам*, который передает IP-пакет не на основе IP-адреса назначения, а на основе поля

метки. При принятии решения о выборе следующего хопа блок продвижения по меткам использует *таблицу коммутации*, которая в стандарте MPLS носит название **таблицы продвижения**. Таблица продвижения в технологии MPLS похожа на аналогичные таблицы других технологий, основанных на технике виртуальных каналов (табл. 4.1).

Таблица 4.1. Пример таблицы продвижения в технологии MPLS

Входной интерфейс	Метка	Следующий хоп	Действия
S0	245	S1	256
S0	27	S2	45
...

Таблицы продвижения для каждого устройства LSR формируются *сигнальным протоколом*, который в MPLS имеет название **протокол распределения меток (Label Distribution Protocol, LDP)**.

Протокол LDP прокладывает через сеть виртуальные каналы, которые в технологии MPLS называют *путями коммутации по меткам (Label Switching Path, LSP)*.

Пути коммутации по меткам

На рис. 4.12 показана сеть MPLS, взаимодействующая с несколькими сетями IP.

Пограничные устройства LSR в технологии MPLS имеют специальное название – **пограничные коммутирующие по меткам маршрутизаторы (Label switch Edge Routers, LER)**.

Устройство LER, являясь функционально более сложным, принимает трафик от других сетей в форме стандартных IP-пакетов, а затем добавляет к нему 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.

LER выполняет отображение IP-адреса на один из существующих в сети путей LSP на основе так называемой **информации о классе эквивалентного продвижения (Forwarding Equivalence Class, FEC)**.

FEC может соответствовать какой-либо подсети IP или же набору IP-подсетей. Принципиально возможно использовать в качестве признаков FEC и отличную от IP-адресов информацию, например, номер интерфейса,

на который пришел пакет, или номер VLAN, если пакет инкапсулирован в кадр Ethernet.

Как и в других технологиях, использующих технику виртуальных каналов, метка имеет *локальное* значение в пределах каждого устройства LER и LSR, то есть при передаче пакета с входного интерфейса на выходной выполняется смена значения метки.

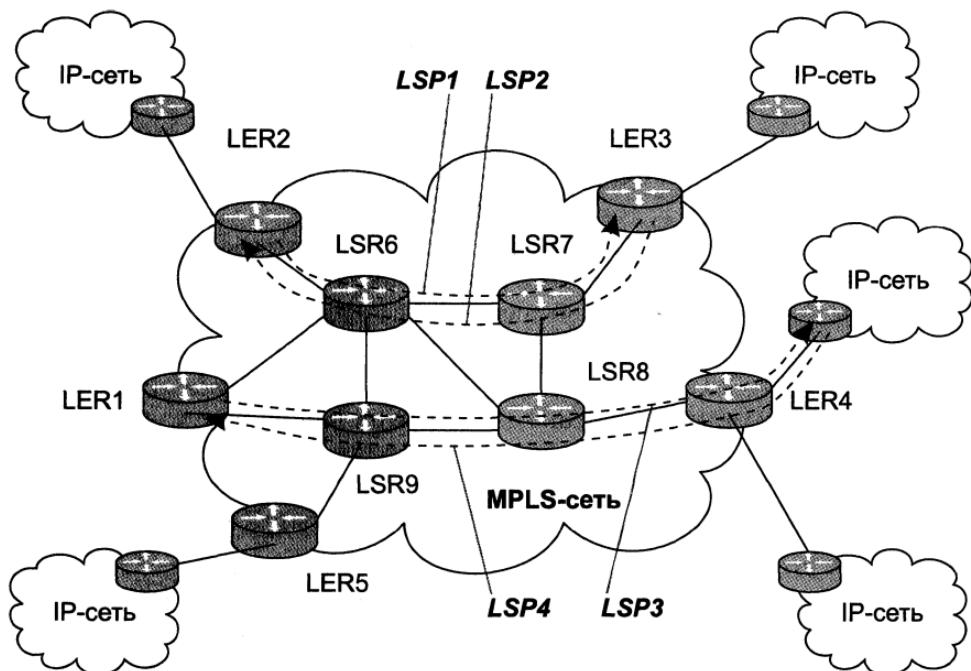


Рис. 4.12. Сеть MPLS

Пути LSP прокладываются в MPLS *предварительно* в соответствии с топологией сети. Существует несколько подходов к прокладке пути, различающихся критериями его выбора и способом нахождения. В качестве двух наиболее популярных критериев выбора пути используются: традиционная метрика, учитывающая номинальную пропускную способность линий связи, и метрика, учитывающая процесс резервирования пропускной способности для потоков данных, проходящих через LSP. Мы вернемся к этому вопросу позже, когда будем рассматривать области применения MPLS.

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. 4.12 показаны две пары путей коммутации по меткам, соединяющие устройства LER2 и LER3, а также LER1 и LER4. Очевидно, что для обеспечения связи между всеми сетями этого недостаточно. Обычно

устройства LER с помощью путей коммутации по меткам должны образовывать полносвязную структуру, которая имеет место в реальных сетях MPLS. Эта структура на рисунке не показана только ввиду громоздкости ее графического представления.

Выходное устройство LER, удалив метку, передает пакет в следующую сеть уже в стандартной для сети IP форме. Таким образом, технология MPLS остается прозрачной для остальных сетей IP.

Заголовок MPLS и технологии канального уровня

Заголовок MPLS состоит из нескольких полей (рис. 4.13).

- *Метка* (20 бит) служит для выбора соответствующего пути коммутации по меткам.
- *Время жизни (TTL)*. Это 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 показателя QoS.
- *Признак дна стека меток – S* (1 бит).

Концепция **стека меток** позволяет организовывать иерархию LSP, когда внутри внешнего пути LSP (называемого также туннелем) существует несколько путей LSP второго уровня; в свою очередь, внутри каждого пути LSP второго уровня может существовать несколько путей LSP третьего уровня и т. д. Мы уже упоминали концепцию иерархии виртуальных каналов, когда изучали технологию ATM, в которой используется два уровня иерархии: виртуальный путь (VP) и виртуальное соединение (VC). MPLS не ограничивает количество уровней иерархии каналов.

Как видно из рисунка, технология MPLS поддерживает несколько типов кадров: PPP, Ethernet, Frame Relay и ATM. Это не означает, что под слоем MPLS работает какая-либо из перечисленных технологий, например Ethernet. Это означает только то, что в технологии MPLS используются форматы кадров этих технологий для помещения в них пакета сетевого уровня, которым почти всегда сегодня является IP-пакет.

В кадрах PPP, Ethernet и Frame Relay заголовок MPLS помещается между оригинальным заголовком и заголовком пакета 3-го уровня. С ячейками ATM технология MPLS поступает по-другому: она пользуется имеющимися полями VPI/VCI в заголовках этих ячеек для меток виртуальных соединений. Поля VPI/VCI служат только для хранения поля метки, остальная часть заголовка MPLS с полями CoS, S и TTL размещается в поле данных ATM-

ячеек и при передаче ячеек коммутаторами ATM, поддерживающими технологию MPLS, не задействуется.

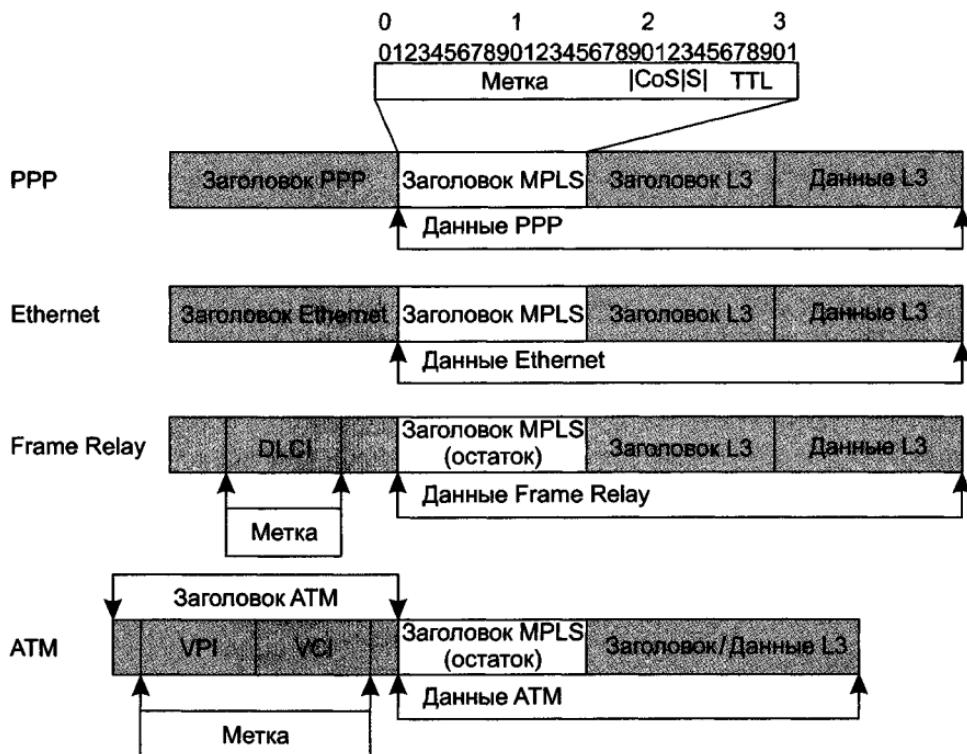


Рис. 4.13. Форматы заголовков нескольких разновидностей технологии MPLS

Отказоустойчивость MPLS

MPLS поддерживает несколько механизмов отказоустойчивости, или в терминах технологии SDH – *автоматического защитного переключения* маршрута в случае отказа какого-либо элемента сети: интерфейса LSR, линии связи или LSR в целом.

Эти механизмы иллюстрирует рис. 4.14, на котором показан основной путь LSP 1, соединяющий устройства LER1 и LER2 через LSR1 и LSR4.

На сегодня стандартизованы три механизма отказоустойчивости в сетях MPLS.

- ❑ **Защита линии.** Такая защита организуется между двумя устройствами LSR, непосредственно соединенными линией связи. Обходной маршрут прокладывается между этими устройствами таким образом, чтобы обойти линию связи в случае ее отказа. На рис. 4.14 таким образом защищена линия связи LSR1-LSR4 за счет установки обходного пути LSP1-2 через

LSR3. Для организации такой защиты применяется иерархия меток: на обходном участке основная метка пути LSR1 проталкивается в стек и используется новый уровень меток пути LSP1-3 до достижения конца обходного маршрута. Затем LSR4 удаляет метку второго уровня и задействует метку основного уровня для продолжения маршрута.

- **Защита узла.** Обходной путь LSP1-3 устанавливается так, чтобы обойти отказалосье устройство LSR, в нашем примере – это устройство LSR4.
- **Защита пути.** В дополнение к основному пути в сети прокладывается путь, связывающий те же устройства LER, но проходящий по возможности через устройства LSR и линии связи, не встречающиеся в основном пути. На рисунке это резервный путь LSP 1-1. Данный механизм самый универсальный, но и самый медленный. Что же касается первых двух, то они по скорости сравнимы с защитой SDH и обеспечивают переключение на уровне примерно 50 мс, за что получили название быстрой перенаправления (fast re-route).

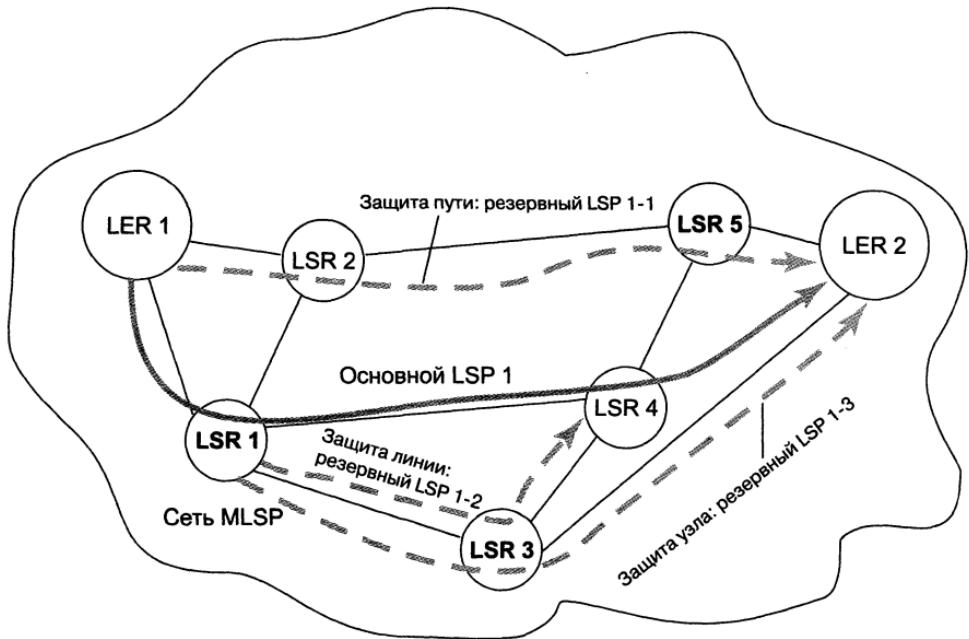


Рис. 4.14. Механизмы отказоустойчивости сетей MPLS

Области применения технологии MPLS

Мы кратко рассмотрели основные принципы, лежащие в основе технологии MPLS. В настоящее время существует несколько областей практического применения MPLS, в которых эти принципы дополняются специфическими механизмами и протоколами, необходимыми для достижения необходимой

функциональности. Далее перечислены те области применения MPLS, которые получили наибольшее распространение.

- ❑ **MPLS IGP.** В данном случае технология MPLS применяется только для ускорения продвижения пакетов сетевого уровня, следующих вдоль маршрутов, выбираемых стандартными внутренними шлюзовыми протоколами маршрутизации (IGP), которые и дали название этой области применения MPLS. Обычно для выбора маршрута применяются протоколы маршрутизации OSPF и IS-IS, а метки между устройствами LSR распределяются протоколом LDP. Эта область соответствует оригинальной идеи создателей технологии, но в настоящее время из-за появления быстродействующих маршрутизаторов утратила свою первоначальную актуальность.
- ❑ **MPLS TE.** В этом случае пути коммутации по меткам выбираются для решения так называемых задач инженеринга трафика (Traffic Engineering, TE) на основе модифицированных протоколов маршрутизации OSPF и IS-IS. В этих протоколах в качестве метрики применяется доступная для резервирования пропускная способность линий связи. Для использования этого режима в каждом запросе на прокладку нового пути LSP нужно указывать требуемую для резервирования за этим путем пропускную способность. Техника MPLS TE не только позволяет обеспечить *рациональную и сбалансированную загрузку всех ресурсов сети* поставщика услуг, но и создает хорошую основу для предоставления транспортных услуг с *гарантированными параметрами QoS*. Для прокладки пути, выбранного протоколами SPF, служит протокол RSVP TE. Возможен также выбор пути внешней системой оптимального планирования маршрутов или полностью ручной выбор.
- ❑ **MPLS VPN.** Эта область применения позволяет поставщику *предоставлять услуги виртуальных частных сетей* (Virtual Private Network, VPN) путем разграничения трафика между пользовательскими сетями. Из самого названия — виртуальная частная сеть — следует, что она каким-то образом воспроизводит свойства *реальной* частной сети. Частная сеть полностью принадлежит одному собственнику (предприятию, компании), она изолирована от других сетей, следовательно, обладает такими качествами, как повышенная безопасность, доступность, предсказуемая пропускная способность, независимость в выборе адресов и имен. Технологии VPN позволяют в сети, совместно используемой несколькими предприятиями (например, в сети провайдера), имитировать сервисы, приближающиеся по качеству к сервисам частной сети. Существует несколько технологий VPN, одна из которых, например, основана на шифровании трафика. В данном варианте MPLS VPN применяется техника виртуальных каналов, принцип образования которых гарантирует получение свойств частной сети в инфраструктуре общедоступной сети с коммутацией пакетов. Это объясняется тем, что в отличие от дейтаграммных сетей, таких как IP (и Интернет, основанный на IP), которые разрешают

каждому узлу взаимодействовать с каждым узлом, сети с виртуальными каналами разрешают взаимодействовать узлам только в том случае, если между ними проложен виртуальный путь. Для организации такого типа VPN широко применялись услуги Frame Relay, сейчас для этих целей используются услуги MPLS.

Глобальные сети IP

Структура глобальной сети IP

Как вы уже знаете, технология IP предназначена для создания составных сетей, при этом в качестве составляющих частей могут выступать как локальные, так и глобальные сети. В последнем случае составная сеть также является глобальной сетью, поэтому имеет смысл говорить о глобальных сетях IP.

В зависимости от того, как устроены слои глобальной сети, находящиеся под уровнем IP, можно говорить и «чистых сетях IP» и о сетях IP «поверх» (over) какой-нибудь технологии, например, IP over ATM. Название «чистая сеть IP» говорит о том, что под уровнем IP не находится никакого другого уровня, выполняющего коммутацию пакетов (кадров или ячеек).

Рисунок 4.15 поясняет такую классификацию. Для предоставления качественных и разнообразных услуг большинство крупных глобальных сетей, особенно сетей коммерческих операторов связи, строится в виде **многослойной сети IP**, схематично показанной на рисунке.

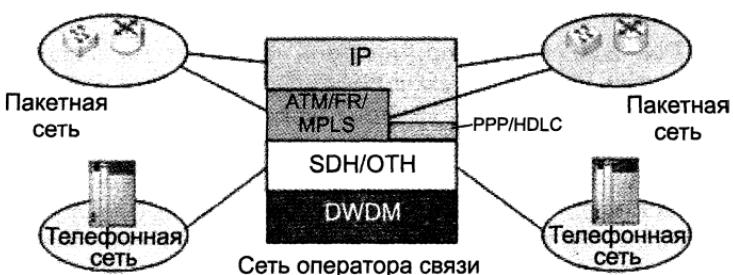


Рис. 4.15. Многоуровневая структура сети оператора связи

Два нижних уровня – это уровни *первой сети*. По классификации семиуровневой модели OSI они соответствуют одному уровню – физическому, так как для пакетной сети первичная сеть выглядит просто как набор физических каналов точка-точка. На самом нижнем уровне первой сети может работать наиболее скоростная на сегодняшний день технология DWDM, образующая спектральные каналы со скоростями 10 Гбит/с и выше. На следующем уровне, поверх DWDM, может применяться технология SDH (с сетью доступа PDH) или OTN, с помощью которой пропускная способность

спектральных каналов делится на менее производительные «мелкие» TDM-подканалы, связывающие интерфейсы коммутаторов пакетной сети (или телефонных коммутаторов). Иногда уровень спектральных каналов DWDM называют нулевым уровнем, а уровень SDH/OTN — первым уровнем, хотя такие названия не являются стандартными.

На основе первичной сети оператор сети может достаточно быстро организовать постоянный цифровой канал между точками подключения оборудования следующего уровня — *наложенной сети* (пакетной или телефонной).

Верхний уровень в приведенной на рисунке модели глобальной сети образован IP-сетью.

Взаимодействие IP с уровнями первичной сети может происходить по двум сценариям. В первом сценарии такое взаимодействие обеспечивает промежуточный слой одной из рассмотренных ранее технологий глобальных сетей, скорее всего, сегодня это будет MPLS, а не ATM или Frame Relay. Такой промежуточный слой, так же как и IP, выполняет коммутацию, но только не пакетов, а кадров или ячеек; сети данного слоя «выглядят» для протокола IP подсетями, которые необходимо объединить в составную сеть.

Принципы и механизмы работы протокола IP поверх технологий глобальных сетей по большей части не отличаются от тех, которые были описаны в главе 3 на примере работы IP поверх Ethernet. Исключением является протокол ARP, который не может выполнять свою работу, то есть автоматически находить соответствие между IP-адресом и адресом глобальной сети (примерами таких адресов является метка виртуального канала Frame Relay или метка LSP технологии MPLS). Причина в том, что технологии глобальных сетей в отличие от Ethernet не поддерживают режима широковещательной передачи кадров. В результате таблицы ARP в случае работы IP поверх Frame Relay, ATM или MPLS формируются вручную.

Второй сценарий получил название «чистой сети IP».

«Чистая сеть IP» отличается от многослойной тем, что под уровнем IP нет другой сети с коммутацией пакетов, такой как ATM или Frame Relay, и IP-маршрутизаторы связываются между собой выделенными каналами (физическими или соединениями OTN/SDH/PDH поверх DWDM).

В такой сети цифровые каналы по-прежнему образуются инфраструктурой двух нижних уровней, а этими каналами непосредственно пользуются интерфейсы IP-маршрутизаторов без какого-либо промежуточного уровня коммутации кадров. В том случае, когда IP-маршрутизатор задействует каналы, образованные в сети SDH/SONET, вариант сети IP получил название **пакетной сети, работающей поверх SONET** (Packet Over SONET, POS).

Однако для того, чтобы маршрутизаторы в модели чистой сети IP могли использовать цифровые каналы, на этих каналах должен работать какой-либо

протокол канального уровня. Такой протокол служит только для упаковки IP-пакетов в кадры, коммутационных способностей от него не требуется, так как протокол обслуживает соединения «точка-точка» между интерфейсами маршрутизаторов. Существует несколько протоколов канального уровня, специально разработанных для таких двухточечных соединений оборудования в глобальной сети.

Из набора существующих двухточечных протоколов протокол IP сегодня использует два: HDLC и PPP.

Протоколы HDLC и PPP

Протокол HDLC (High-level Data Link Control — высокоуровневое управление линией связи) представляет целое семейство протоколов, реализующих функции канального уровня.

Первое, что мы отметим по поводу протокола HDLC, — его *функциональное разнообразие*. Он может работать в нескольких весьма отличающихся друг от друга режимах, поддерживает не только двухточечные соединения, но и соединения с одним источником и несколькими приемниками, он также предусматривает различные функциональные роли взаимодействующих станций. Сложность HDLC объясняется тем, что это очень «старый» протокол, разработанный еще в 70-е годы для ненадежных каналов связи. Поэтому в одном из режимов протокол HDLC подобно протоколу TCP поддерживает процедуру установления логического соединения и процедуры контроля передачи кадров, а также восстанавливает утерянные или поврежденные кадры. Существует также и дейтаграммный режим работы HDLC, в котором логическое соединение не устанавливается, а кадры не восстанавливаются.

В IP-маршрутизаторах чаще всего используется версия протокола HDLC, разработанная компанией Cisco. Несмотря на то что эта версия является фирменным протоколом, она стала стандартом де-факто для IP-маршрутизаторов большинства производителей. Версия Cisco HDLC работает только в дейтаграммном режиме, что соответствует современной ситуации с незашумленными надежными каналами связи. По сравнению со стандартным протоколом версия Cisco HDLC включает несколько расширений, главным из которых является многопротокольная поддержка. Это означает, что в заголовок кадра Cisco HDLC добавлено поле типа протокола, подобное полю EtherType и содержащее код протокола, данные которого переносит кадр Cisco HDLC. В стандартной версии HDLC такое поле отсутствует.

Протокол PPP (Point-to-Point Protocol) является стандартным протоколом Интернета.

Особенностью протокола PPP, отличающей его от других протоколов канального уровня, является *гибкая и многофункциональная процедура* принятия параметров соединения. Стороны обмениваются различными параметрами, такими как качество линии, размер кадров, тип протокола аутентификации и тип инкапсулируемых протоколов сетевого уровня.

В компьютерной сети конечные системы часто различаются размерами буферов для временного хранения пакетов, ограничениями на размер пакета, списком поддерживаемых протоколов сетевого уровня. Физическая линия, связывающая конечные устройства, может варьироваться от низкоскоростной аналоговой до высокоскоростной цифровой линии с различными уровнями качества обслуживания.

Протокол, который используется в PPP для переговоров по поводу принятия параметров соединения, называется **протоколом управления линией связи** (Link Control Protocol, LCP). Чтобы справиться со всеми возможными ситуациями, в протоколе PPP имеется набор стандартных установок, действующих по умолчанию и учитывающих все стандартные конфигурации. При установлении соединения два взаимодействующих устройства для нахождения взаимопонимания пытаются сначала использовать эти установки. Каждый конечный узел описывает свои возможности и требования. Затем на основании этой информации принимаются параметры соединения, устраивающие обе стороны. Переговорная процедура протоколов может и не завершиться соглашением о каком-нибудь параметре. Если, например, один узел предлагает в качестве MTU значение 1000 байт, а другой отвергает это предложение и, в свою очередь, предлагает значение 1500 байт, которое отвергается первым узлом, то по истечении тайм-аута переговорная процедура может закончиться безрезультатно.

Одним из важных параметров PPP-соединения является *режим аутентификации*. Для целей аутентификации PPP предлагает по умолчанию *протокол аутентификации по паролю* (PAP), передающий пароль по линии связи в открытом виде, или *протокол аутентификации по квитированию вызова* (CHAP), не передающий пароль по линии связи и поэтому обеспечивающий более высокий уровень безопасности сети. Пользователям также разрешается добавлять новые алгоритмы аутентификации. Кроме того, пользователи могут влиять на выбор алгоритмов компрессии заголовка и данных.

Несмотря на то что протокол PPP работает в режиме установления соединения, контролем доставки кадров и их восстановлением он не занимается, так как во время разработки протокола надежные цифровые каналы уже доминировали в телекоммуникационных сетях.

Carrier Ethernet – Ethernet операторского класса

Название **Carrier Ethernet** является сокращением от Carrier Grade Ethernet, то есть это Ethernet операторского класса. В наиболее широком смысле под Carrier Ethernet понимают как услуги Ethernet, которые операторы связи предоставляют в глобальном масштабе, так и технологии, лежащие в основе этих услуг.

Движущие силы Carrier Ethernet

Как мы знаем, классическая технология Ethernet создавалась исключительно как технология локальных сетей, и до недавнего времени сети этого класса и были единственной областью ее применения. Но бесспорный успех технологии Ethernet в локальных сетях, откуда она вытеснила все остальные технологии, привел к напрашивающейся идеи о ее использовании и в глобальных сетях (которые и являются по большей части операторскими).

Потенциальных преимуществ от экспансии Ethernet за пределы локальных сетей несколько. Для пользователей это означает возможность соединения своих территориально рассредоточенных сетей тем же способом, к которому они привыкли в офисных сетях, то есть на уровне коммутаторов Ethernet без привлечения IP. Другим важным преимуществом, причем как для пользователя, так и для провайдера, является относительно низкая стоимость оборудования Ethernet. Порты Ethernet всегда обладали самой низкой стоимостью по сравнению с портами любой другой технологии (естественно, при учете скорости передачи данных по портом). Низкая стоимость изначально была результатом простоты технологии Etherjet, которая предлагает только минимальный набор функций по передаче кадров. Низкая стоимость оборудования Ethernet при удовлетворительной функциональности привела к доминированию Ethernet на рынке оборудования локальных сетей, ну а далее начал работать механизм положительной обратной связи: хорошие продажи – массовое производство – еще более низкая стоимость и т. д.

Стремление к унификации также является движущей силой экспансии Ethernet в глобальные сети. Сетевой уровень уже давно демонстрирует однородность, так как на нем теперь работает только протокол IP; 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 локальном виде не готова. С одной стороны, для того чтобы успешно действовать в сетях операторов связи, технология и воплощающее ее оборудование должны обладать определенными характеристиками, в первую очередь такими, как надежность, отказоустойчивость,

масштабируемость и управляемость. Поэтому работа над наделением Ethernet подобными свойствами уже началась. С другой стороны, услуга Carrier Ethernet в сети провайдера может быть реализована средствами других, отличных от Carrier Ethernet, технологий (рис. 4.16).

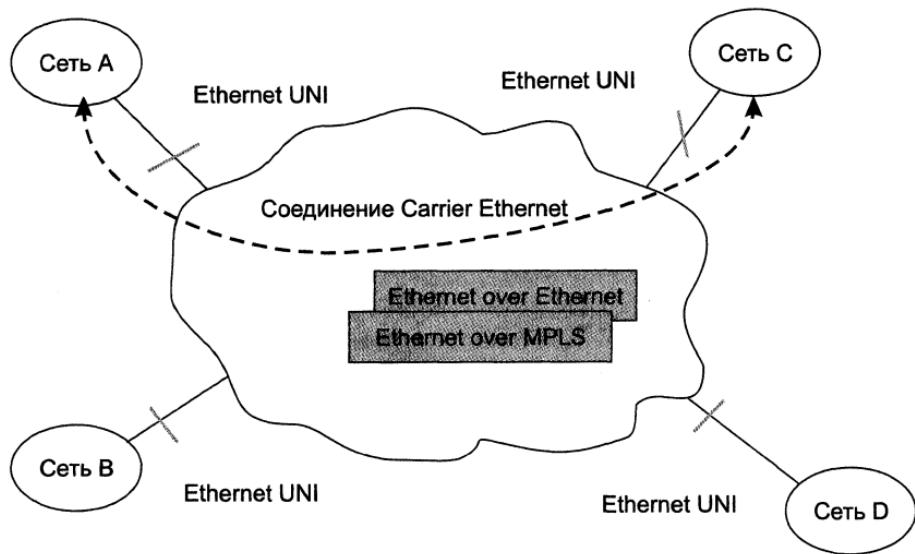


Рис. 4.16. Различные варианты реализации услуги Carrier Ethernet

Независимо от выбранного варианта, для пользователя глобальная услуга Ethernet всегда предоставляется с помощью набора стандартных интерфейсов Ethernet (Ethernet UNI) на каналах доступа к сети провайдера. Эти интерфейсы поддерживают одну из спецификаций Ethernet физического уровня, например, 100Base-FX или 1000Base-LX, а также стандартные кадры Ethernet. Однако если внешние услуги Carrier Ethernet от разных провайдеров выглядят более-менее однотипно (естественно, различия могут существовать, например, не все провайдеры гарантируют параметры качества обслуживания, некоторые предоставляют только соединения «точка-точка» и т. п.), то внутренняя организация такой услуги в пределах сети провайдера может отличаться значительно.

Ethernet на основе MPLS

В этом варианте туннели MPLS (с некоторой надстройкой) используются как основной транспортный механизм провайдера, позволяющий эмулировать услугу Ethernet для клиентов. Такие свойства MPLS, как поддержка детерминированных маршрутов, наличие механизма быстрой переключения, обеспечивающего быстрое (сравнимое с SDH) переключение с основного маршрута на резервный, развитые средства контроля работоспособности

соединения, сделали эту технологию привлекательной опцией для операторов связи. Кроме того, MPLS – это весьма зрелая технология с более чем 10-летней историей; она применяется сегодня в магистральных сетях очень многих крупных провайдеров связи для различных целей, так что ее надежность и эффективность проверены практикой. Комитет IETF, занимающийся разработкой стандартов MPLS, выпустил несколько документов RFC, описывающих детали процесса эмуляции Ethernet с помощью этой технологии, которая получила название Ethernet на основе MPLS (Ethernet over MPLS, EoMPLS).

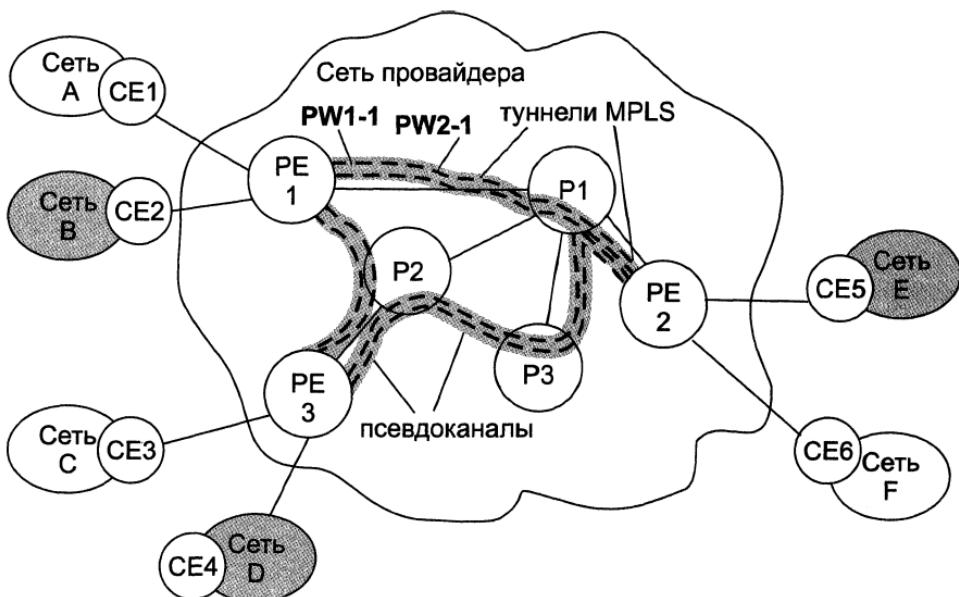


Рис. 4.17. Услуга VPLS

На рис. 4.17 показана организация сервиса виртуальных локальных сетей (Virtual Private LAN Services, VPLS) с помощью EoMPLS. При предоставлении услуги VPLS сеть провайдера выглядит для клиентов как локальная сеть, передающая кадры между сайтами клиента на основе MAC и, при желании, учитывая значения тегов VLAN в этих кадрах. Сеть провайдера на рисунке оказывает две услуги двух частных виртуальных сетей: сеть VPLS1 объединяет сети A, C и F, а VPLS2 – сети B, D и E. Сети клиентов присоединяются к сети провайдера с помощью своих пограничных устройств CE1, 2, ..., 6. Сети, принадлежащие к одной сети VPLS, могут общаться между собой, но не могут общаться с сетями, принадлежащими другой сети VPLS.

Услуга VPLS реализуется провайдером с помощью особого типа путей LSP в сети MPLS, называемых псевдоканалами. Псевдоканалы образуют логические соединения между пограничными маршрутизаторами провайдера (PE 1, PE 2 и PE 3 на рисунке, или LER в терминологии MPLS) по топологии

«каждый с каждым». Пограничные маршрутизаторы используют специальный набор псевдоканалов для передачи кадров каждой отдельной сети VPLS. Так как в нашем примере провайдер эмулирует услуги двух сетей VPLS, то между его пограничными маршрутизаторами существует два набора псевдоканалов, PW1-N и PW2-N, показанных разными типами пунктирных линий.

Все наборы псевдоканалов проходят внутри одних и тех же туннелей, соединяющих пограничные маршрутизаторы (тунNELи показаны сплошной толстой линией). Фактически все требуемые свойства услуг сетей операторского класса – детерминированность маршрута, гарантии пропускной способности, отказоустойчивость – обеспечиваются туннелями.

Посмотрим, как работают пограничные маршрутизаторы при обработке пользовательских кадров. Пусть маршрутизатор PE1 получает кадр от сети A с адресом назначения в сети F. Так как сеть A приписана к сети VPLS1, то PE1 продвигает кадр посредством набора псевдоканалов PW1-N. Для того чтобы решить, какой из псевдоканалов из этого набора задействовать, пограничные маршрутизаторы, подобно мостам, изучают пользовательские MAC-адреса. Поэтому, если адрес назначения уже изучен маршрутизатором PE1, то есть пакет с таким обратным адресом уже проходил через этот маршрутизатор, он выбирает псевдоканал, ведущий к PE2; в противном случае PE1 затапливает набор псевдоканалов PW1-N этим кадром.

Услуги VPLS приобрели большую популярность и оказываются сегодня практически всеми крупными провайдерами.

Ethernet на основе Ethernet, или Carrier Ethernet Transport

Если классическая технология Ethernet недостаточно хороша для работы в сетях операторов связи, то, может быть, ее можно усовершенствовать? По этому пути пошли многие производители коммуникационного оборудования, их усилия в этой области стандартизует IEEE. Работы по преобразованию Ethernet в технологию операторского класса идут в нескольких областях: наделение Ethernet функциями контроля ошибок (Ethernet OAM); разделение адресных пространств провайдера и пользователя (иначе коммутаторы провайдера окажутся перегруженными обработкой адресов многочисленных пользовательских узлов); контроль над маршрутами трафика; резервирование маршрутов.

Одним из перспективных стандартов, позволяющих провайдерам использовать в своей сети Ethernet, является стандарт магистральных мостов провайдера (Provider Backbone Bridges, PBB), принятый IEEE летом 2008. Стандарт PBB разделяет адресные пространства пользователей и провайдера за счет того, что пограничные коммутаторы провайдера инкапсулируют пользовательские кадры Ethernet в новые кадры Ethernet, которые применяются только в пределах сети провайдера для доставки пользовательских кадров.

до выходного пограничного коммутатора. Адресами провайдерских кадров Ethernet служат MAC-адреса пограничных коммутаторов (Backbone Edge Bridges, BEB). По сути, в сети провайдера работает независимая иерархия Ethernet со своими MAC-адресами и делением сети на виртуальные локальные сети (VLAN) так, как это удобно провайдеру. Из-за наличия двух уровней MAC-адресов в кадрах провайдера стандарт PBB получил также название MAC-in-MAC.

На рис. 4.18 показана сеть провайдера, оказывающая услуги Ethernet своим клиентам на основе стандарта PBB. Так же, как и в примере из предыдущего раздела, провайдер предоставляет услуги двух частных виртуальных сетей, одна из которых (VPLS1) объединяет сети C1, C3 и C5, а другая (VPLS2) – сети C2, C4 и C6.

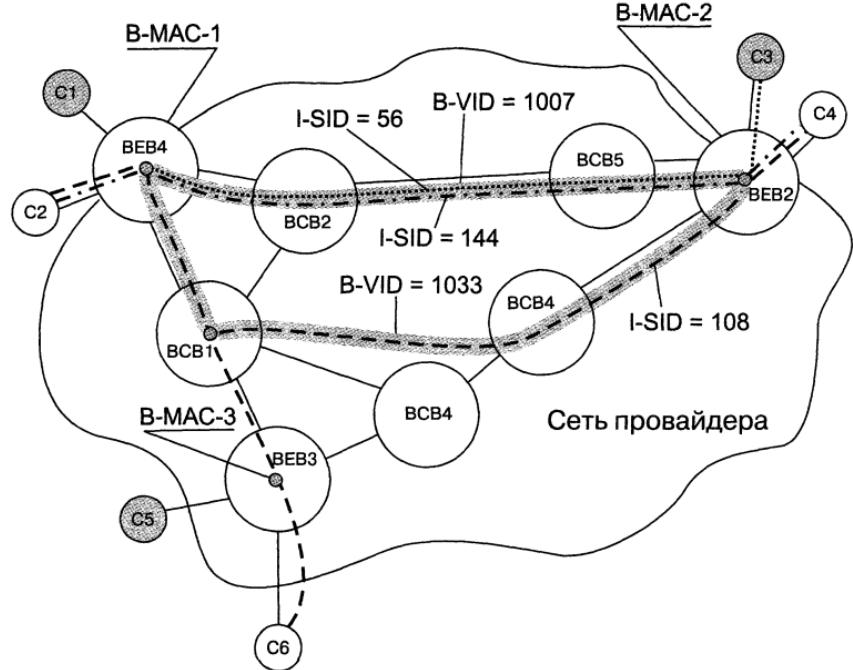


Рис. 4.18. Организация PBB

В сети провайдера организовано две различные магистральные *виртуальные локальные сети* (B-VLAN) с идентификаторами 1007 и 1033 (обозначены как B-VID 1007 и B-VID 1033 соответственно). Однако ошибочно было бы полагать, что эти различные сети прямо соответствуют двум пользовательским сетям VPLS, то есть служат для разделения пользовательского трафика. Нет, магистральные сети VLAN применяются в целях агрегированного управления трафиком в сети провайдера, создавая транспортную основу с поддержанием, например, гарантий пропускной способности и отказоустойчивости для пользовательских потоков. Идея здесь та же, что и в случае туннелей

MPLS как транспортной основы услуги VPLS. В нашем примере различные сети B-VLAN призваны поддерживать трафик разного типа. B-VLAN 1007 поддерживает более требовательный голосовой трафик, а B-VLAN 1033 — менее требовательный эластичный трафик данных. В соответствии с этим назначением созданы и два покрывающих дерева для каждой из сетей.

Для разделения пользовательского трафика стандарт PBB ввел в заголовок провайдерского кадра Ethernet еще один тег — тег идентификатора сервиса I-SID. Идентификатор сервиса I-SID представляет собой трехбайтное поле, которое может кодировать до 16 миллионов пользовательских сервисов, например, таких как VPLS. В то время как идентификатор B-VID совместно с адресами B-MAC применяется всеми коммутаторами сети провайдера для продвижения кадров, идентификаторы I-SID задействуют только пограничные коммутаторы провайдера для отделения трафика различных пользовательских сервисов, в нашем примере — для отделения трафика VPLA1 от трафика VPLA2. Можно сказать, что идентификаторы сервисов стандарта PBB играют ту же роль, что и псевдоканалы в стандарте EoMPLS.

В нашем примере представлены три пользовательских сервиса, помеченные как I-SID 56, 144 и 108. Сервисы 56 (трафик от сети C1) и 144 (трафик от сети C2 со значением 305 пользовательской виртуальной локальной сети) отображаются пограничным коммутатором BEB1 на B-VLAN 1007, так как эти сервисы переносят пользовательский голосовой трафик, а данная сеть D-VLAN была создана для этого типа трафика. В то же время сервис I-SID 108 (приходящий от сети C2 со значением 500 пользовательской виртуальной локальной сети) отображается на B-VLAN 1033, так как сервис 108 переносит эластичный пользовательский трафик данных. Отображение пользовательского трафика на соответствующую магистральную виртуальную локальную сеть и соответствующий сервис I-SID выполняется администратором сети при конфигурировании пограничных коммутаторов BEB.

Выходной пограничный коммутатор BEB2 извлекает из поступающих провайдерских кадров пользовательские кадры и на основе значения I-SID передает их в конкретную сеть пользователя. Пограничные коммутаторы изучают MAC-адреса пользователей, чтобы определить магистральный адрес назначения B-MAC в тех случаях, когда пользовательский сервис реализован по топологии «каждый с каждым» (сервис I-SID 108 в нашем примере).

Про популярность услуг Carrier Ethernet на основе стандарта PBB пока говорить рано, время покажет, насколько он окажется востребованным. В стандарте PBB отказоустойчивость обеспечивается традиционным для Ethernet способом — с помощью протокола покрывающего дерева. В настоящее время IEEE работает над версией стандарта PBB под названием PBB TE (от Traffic Engineering — инжиниринг трафика), в котором отказоустойчивость будет поддерживаться за счет резервных путей с детерминированным маршрутом.

Удаленный доступ

Проблемы удаленного доступа

Термин **удаленный доступ** (remote access) часто применяют, когда речь идет о доступе пользователя домашнего компьютера к Интернету или сети предприятия, которая находится от него на значительном расстоянии и поэтому обязательно приходится действовать через глобальные связи. В последнее время под удаленным доступом стали понимать доступ не только изолированных компьютеров, но и домашних сетей, объединяющих несколько компьютеров членов семьи. Такими же небольшими сетями располагают малые офисы предприятий, насчитывающие 2–3 сотрудника.

Организация удаленного доступа является одной из наиболее острых проблем компьютерных сетей в настоящее время. Она получила название «проблемы последней мили», где под последней мильей подразумевается расстояние от **точки присутствия** (Point Of Presence, POP) оператора связи до помещений клиентов. Сложность этой проблемы определяется несколькими факторами. С одной стороны, современным пользователям необходим высокоскоростной доступ, обеспечивающий качественную передачу трафика любого типа: данных, голоса, видео. Для этого нужны скорости несколько мегабитов или, по крайней мере, несколько сотен килобитов в секунду. С другой стороны, подавляющее большинство домов в больших и малых городах и особенно в сельской местности по-прежнему соединены с точками присутствия операторов связи абонентскими окончаниями телефонной сети, изначально не предназначавшихся для передачи данных.

Кардинальная перестройка кабельной инфраструктуры доступа вряд ли возможна в ближайшее время, слишком масштабна эта задача из-за огромного количества зданий и домов, географически рассеянных по огромной территории. И хотя в некоторых странах в последнее время стали прокладывать к домам высокоскоростные оптические линии, таких стран не так уж много, да и этот процесс затронул пока только большие города и крупные здания с множеством потенциальных пользователей.

Долгое время наиболее распространенной технологией доступа был **коммутируемый доступ**, когда пользователь устанавливал коммутируемое соединение с корпоративной сетью или Интернетом через телефонную сеть с помощью модема. Такой способ обладает существенным недостатком — скорость доступа ограничена несколькими десятками килобитов в секунду из-за фиксированной узкой полосы пропускания примерно в 3,4 кГц, выделяемой каждому абоненту телефонной сети. Такие скорости сегодня все меньше и меньше устраивают пользователей.

В настоящее время для организации скоростного удаленного доступа привлекаются различные технологии, в которых используется существующая инфраструктура абонентских окончаний — телефонные сети или сети

кабельного телевидения. После достижения POP поставщика телефонных услуг или услуг кабельного телевидения по такому окончанию компьютерные данные уже не следуют по телефонной сети или сети кабельного телевидения, а ответвляются с помощью специального оборудования в компьютерную сеть передачи данных. Это позволяет преодолеть ограничения на полосу пропускания, отводимую абоненту в телефонной сети или сети кабельного телевидения, и повысить скорость доступа.

Наиболее популярной технологией такого типа является **технология ADSL** (Asymmetric Digital Subscriber Line — асимметричная цифровая абонентская линия), в которой телефонные абонентские окончания и кабельные модемы работают поверх сети кабельного телевидения. Эта технология обеспечивает скорость от нескольких сотен килобитов до нескольких мегабитов в секунду. Принципиальным отличием модемов ADSL от коммутируемых модемов является то, что модемы ADSL передают информацию только на сравнительно коротком («последняя миля») абонентском окончании, которое в зависимости от типа кабеля имеет полосу пропускания примерно в 1 МГц. В точке присутствия, к которой ведет абонентское окончание, установлены *мультиплексоры сети передачи данных*, ответвляющие сигналы модема ADSL от сигналов телефона и направляющие данные в компьютерную сеть провайдера, то есть телефонная сеть в этом варианте совсем не используется, задействовано только ее абонентское окончание. В то же время коммутируемые модемы работают через телефонную сеть, так как серверы доступа провайдера в этом случае расположены чаще всего не в точке присутствия абонента, а в некоторой центральной точке присутствия телефонного оператора. Отсюда и возникает ограничение в 3,4 кГц, так как сигнал коммутируемого модема проходит через телефонные коммутаторы.

Применяются также различные **беспроводные технологии доступа**, обеспечивающие как фиксированный, так и мобильный доступ. Набор применяемых беспроводных технологий очень широк, в него входят и беспроводные сети Ethernet (802.11), и различные фирменные технологии, и технологии передачи данных по сети мобильной телефонии, и технологии фиксированного доступа, например, нового стандарта 802.16.

Схемы удаленного доступа

Рисунок 4.19 иллюстрирует разнообразный и пестрый мир удаленного доступа. Мы видим здесь клиентов различных типов, различающихся применяемым оборудованием и требованиями к параметрам доступа. Кроме того, помещения клиентов могут быть соединены с ближайшей точкой доступа оператора связи (то есть с ближайшим центральным офисом, если пользоваться терминологией операторов телефонной сети) различными способами: с помощью аналогового или цифрового окончания телефонной сети, телевизионного кабеля, беспроводной связи. Наконец, сам оператор связи может иметь различную специализацию, то есть быть либо поставщиком

телефонных услуг, либо поставщиком услуг Интернета, либо оператором кабельного телевидения или же быть универсальным оператором, предоставляющим весь спектр услуг и обладающим собственными сетями всех типов.

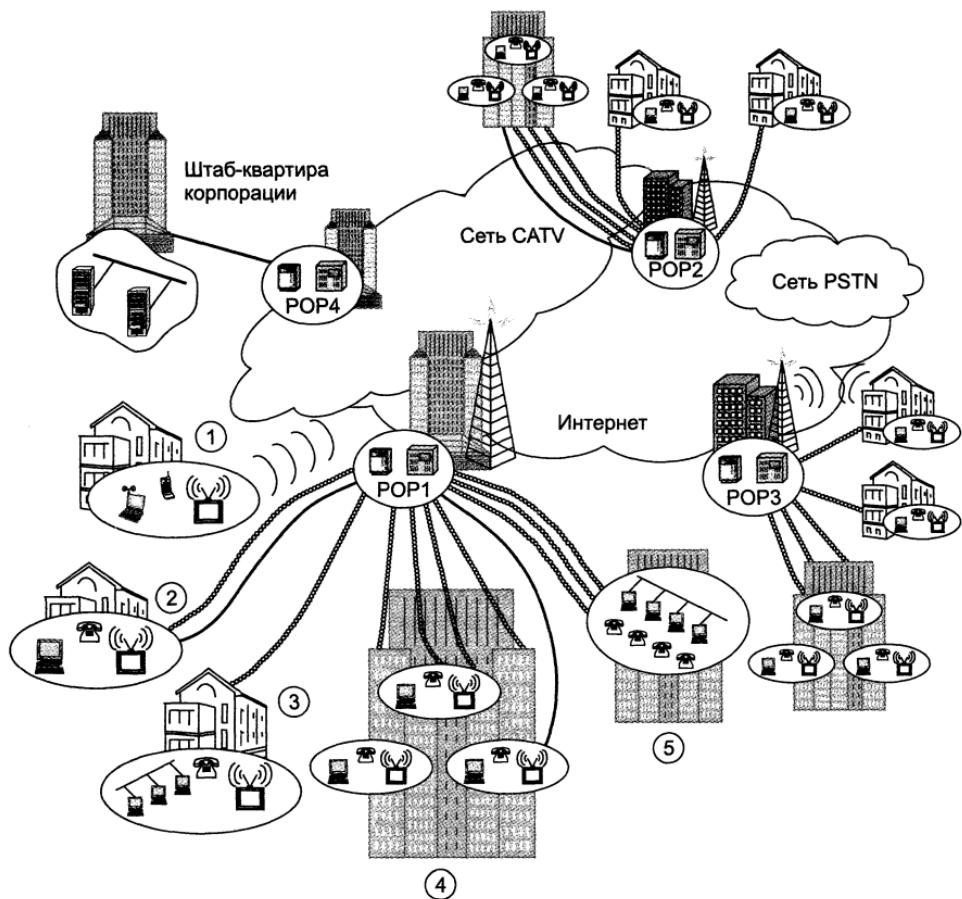


Рис. 4.19. Клиенты удаленного доступа

Рассмотрим каждый элемент представленной схемы доступа более подробно.

Клиенты 1 и 2 являются наиболее типичными пользователями, так как каждый из них имеет только один компьютер, которому необходимо обеспечить доступ к удаленной компьютерной сети. Помимо компьютера эти клиенты пользуются телефоном и телевизором, поэтому абонентские окончания этих устройств можно задействовать для организации доступа компьютера к сети передачи данных.

Клиент 2 пользуется двумя кабельными абонентскими окончаниями: традиционным аналоговым телефонным на основе витой пары и коаксиальным

телеизионным кабелем кабельного телевидения. Эти абонентские окончания обладают существенно разными характеристиками. Так, витая пара при расстоянии 1–2 км между помещением клиента и POP поставщика услуг обычно имеет полосу пропускания примерно несколько мегагерц, в то время как коаксиальный кабель обеспечивает полосу пропускания в несколько десятков мегагерц.

У клиента 1 отсутствуют проводные абонентские окончания, так как он пользуется мобильным телефоном, кроме того, он не является клиентом кабельного телевидения, принимая телевизионный сигнал только по воздуху.

Таким образом, с целью организации удаленного доступа для клиента 2 поставщик услуг может использовать либо существующее телефонное абонентское окончание, либо телевизионный кабель. Для клиента 1 такой возможности нет, поэтому поставщик услуг должен предоставить ему беспроводную связь или же проложить новый кабель между его домом и ближайшей точкой присутствия.

Отличительной особенностью клиентов 1 и 2 является несимметричный характер трафика, так как домашние пользователи в основном загружают информацию на свой компьютер в процессе путешествий по Интернету. Ответом на такие потребности являются асимметричные технологии, такие как ADSL, которые поддерживают существенно большую скорость для загрузки информации на компьютер пользователя, чем выгрузки на удаленный сервер.

Клиент 3 отличается от двух предыдущих тем, что имеет несколько компьютеров, объединенных в локальную сеть. Таким клиентом может быть как частное лицо, так и небольшой офис предприятия. Удаленный доступ для локальной сети отличается повышенными требованиями к пропускной способности. Кроме того, трафик может иметь симметричный характер, если домашняя сеть включает сервер, поставляющий информацию пользователям Интернета или сотрудникам других офисов предприятия. Так как клиент 3 не имеет кабельного окончания сети CATV (cable TV), то ему можно обеспечить доступ только по телефонному окончанию. Клиент 3 может организовать свою сеть IP различными способами. Он может попросить у поставщика услуг пул IP-адресов, чтобы каждый его компьютер имел отдельный публичный постоянный IP-адрес. Это наиболее гибкий вариант для клиента, так как в этом случае каждый его компьютер может быть полноправным узлом Интернета и исполнять роль не только клиентской машины, но и сервера с зарегистрированным доменным именем. Очевидно, что в этом случае локальная сеть клиента должна иметь пограничный маршрутизатор, через который осуществлять связь с сетью поставщика услуг. Другой вариант организации сети IP может быть основан на использовании техники NAT (Network Address Translator).

Клиенты 4 являются жителями многоквартирного дома, который соединен с POP многочисленными витыми парами телефонных абонентских

окончаний (по одной для каждой квартиры), а также кабелем САТВ. Наличие одного кабеля САТВ для большого количества клиентов создает дополнительные проблемы в плане организации доступа, так как кабель в этом случае является разделяемой средой. Применение телефонных абонентских окончаний для удаленного доступа жителей многоквартирного дома ничем не отличается от подключения отдельного абонента (клиента 2).

Клиенты 5 также являются жильцами многоквартирного дома, но в этом доме поставщик услуг развернул локальную сеть. К этой локальной сети подключаются компьютеры тех жильцов дома, которые решили стать абонентами данного поставщика услуг. Такой вариант эффективен для поставщика услуг при достаточно большом количестве абонентов в доме. Локальная сеть многоквартирного дома требует более высоких скоростей доступа, чем отдельные компьютеры или домашние сети индивидуальных клиентов, поэтому поставщик услуг должен задействовать абонентское окончание с широкой полосой пропускания — для этой цели может быть использован существующий кабель САТВ, специально проложенный коаксиальный кабель Ethernet или заново проложенный оптический кабель.

Поставщик услуг удаленного доступа может обслуживать клиентов всех типов или же специализироваться на каком-то определенном типе клиентов, например, жителях частных или многоквартирных домов, работниках небольших офисов. Универсальный поставщик услуг доступа должен поддерживать любые варианты организации «последней мили», что усложняет его оборудование и применяемые технологии доступа.

В любом случае для передачи данных по какому-либо абонентскому окончанию поставщик услуг должен обеспечить для этого окончания передачу компьютерных данных и совместить эту передачу с передачей информации, для которой это окончание было спроектировано, например, с аналоговой телефонной информацией или с сигналом кабельного телевидения. Затем на основе этих средств физического уровня поставщик услуг должен предоставить клиенту тот или иной вариант сервиса доступа.

Наиболее простой вариант доступа в Интернет предоставляет клиенту **незащищенное соединение** с серверами корпоративной сети, что грозит плохими последствиями. Во-первых, конфиденциальные данные, передаваемые по Интернету, могут быть перехвачены или искажены. Во-вторых, при таком способе администратору корпоративной сети трудно ограничить доступ к своей сети несанкционированных пользователей, так как IP-адреса легальных пользователей (сотрудников предприятия) заранее неизвестны. Поэтому предприятия предпочитают пользоваться защищенным доступом, основанным на технологии виртуальных частных сетей (VPN).

ГЛАВА 5. Сетевые услуги

Малые услуги, оказываемые вовремя, являются величайшими благодеяниями для тех, кто их получает.

Демокрит

С точки зрения пользователей компьютерные сети представляют собой набор услуг, таких как электронная почта, WWW, интернет-телефония и интернет-телевидение. Транспортные функции сети, обеспечивающие работу этих услуг, скрыты от пользователей, хотя иногда и влияют на некоторые детали предоставления услуги, например, недостаточно высокая надежность доступа в Интернет по телефонным каналам повлияла на использование коротких сеансов TCP в службе WWW при передаче содержания веб-страниц. Помимо услуг, ориентированных на конечных пользователей, существуют также услуги, ориентированные на администраторов сети, решавших задачи конфигурирования и управления сетевыми устройствами; в эту категорию услуг входят услуги FTP, telnet и SNMP. Дополняют общую картину уже рассмотренные нами служебные услуги, помогающие компьютерам и сетевым устройствам организовать свою работу, такие как услуги служб DNS и DHCP.

Электронная почта

Сетевая почтовая служба (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 он 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, другая часть (тело сообщения) — это собственно то «письмо», которое предназначается для прочтения, прослушивания или просмотра адресатом (RFC 8022).

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 шифрации, о срочности доставки, о необходимости подтверждения факта прочтения этого сообщения адресатом и др. Дополнительная информация заголовка может оповещать почтового клиента получателя об использовании той или иной кодировки. Помимо основной кодировки ASCII, современные почтовые системы позволяют создавать сообщения, включающие изображения (в форматах GIF и JPEG), а также аудио- и видеофайлы.

Протокол SMTP

В качестве средств передачи сообщения почтовая служба использует стандартный, разработанный специально для почтовых систем протокол SMTP (Simple Mail Transfer Protocol — простой протокол передачи почты). Как и большинство других протоколов прикладного уровня, SMTP реализуется несимметричными взаимодействующими частями: SMTP-клиентом

и SMTP-сервером. Важно отметить, что этот протокол ориентирован на передачу данных по направлению от клиента к серверу, следовательно, SMTP-клиент работает на стороне отправителя, а SMTP-сервер — на стороне получателя. SMTP-сервер должен постоянно быть в режиме подключения, ожидая запросов со стороны SMTP-клиента.

Логика протокола SMTP является действительно достаточно простой (как это и следует из его названия). После того как, применяя графический интерфейс своего почтового клиента, пользователь щелкает на значке, инициирующем отправку сообщения, SMTP-клиент посылает запрос на установление TCP-соединения на порт 25 (это назначенный порт SMTP-сервера). Если сервер готов, то он посыпает свои идентифицирующие данные, в частности свое DNS-имя. Затем клиент передает серверу адреса (имена) отправителя и получателя. Если имя получателя соответствует ожидаемому, то после получения адресов сервер дает согласие на установление TCP-соединения, и в рамках этого надежного логического канала происходит передача сообщения. Используя одно TCP-соединение, клиент может передать несколько сообщений, предваряя каждое из них указанием адресов отправителя и получателя. После завершения передачи TCP- и SMTP-соединения разрываются. Если в начале сеанса связи SMTP-сервер оказался не готов, то он посыпает соответствующее сообщение клиенту, и тот снова посыпает запрос, пытаясь заново установить соединение. Если сервер не может доставить сообщение, то он передает отчет об ошибке отправителю сообщения и разрывает соединение. После того как передача сообщения благополучно заканчивается, переданное сообщение сохраняется в буфере на сервере.

ПРИМЕЧАНИЕ

Хотя в любом протоколе предполагается обмен данными между взаимодействующими частями, то есть данные передаются в обе стороны, различают протоколы, ориентированные на передачу (pull protocols), и протоколы, ориентированные на прием данных (push 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

Теперь, когда мы обсудили основные составляющие почтовой службы, давайте рассмотрим несколько основных схем ее организации. Начнем с простейшего, практически не используемого сейчас варианта, когда отправитель непосредственно взаимодействует с получателем. Как показано на рис. 5.1, у каждого пользователя на компьютере установлены почтовые клиент и сервер.

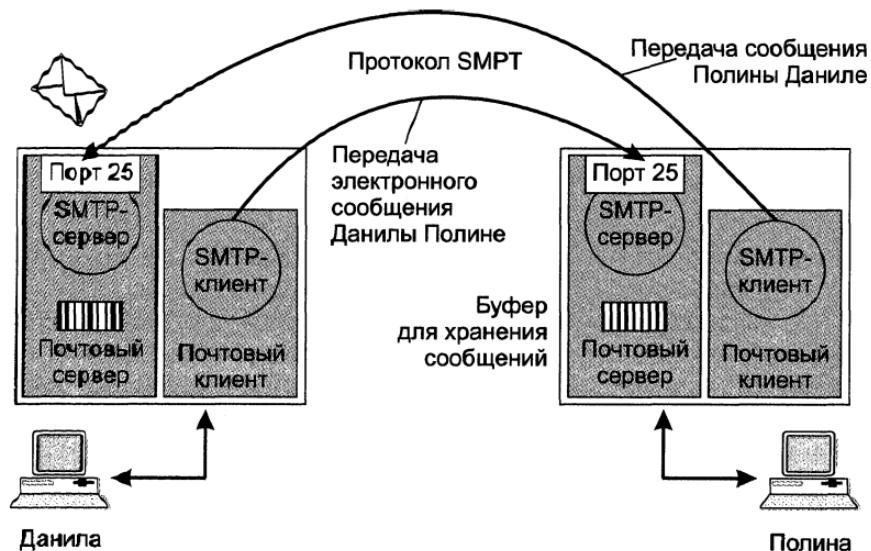


Рис. 5.1. Схема непосредственного взаимодействия клиента и сервера

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 не только пользоваться 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, и 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. Понятно, что в том случае, когда Полина решит направить электронное сообщение Даниле, схема работы почтовой службы будет симметричной.

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. Это означает, что для того, чтобы письма, направленные Полине, доходили до нее, ее компьютер должен постоянно находиться в режим 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, в которых хранятся DNS-имена почтовых серверов, обслуживающих пользователей, относящихся к этому домену.

На рис. 5.2 представлена схема с выделенным почтовым сервером. Чтобы не усложнять рисунок, мы показали на нем только те компоненты, которые участвуют в передаче сообщения от Данилы к Полине. Для обратного случая схема должна быть симметрично дополнена.

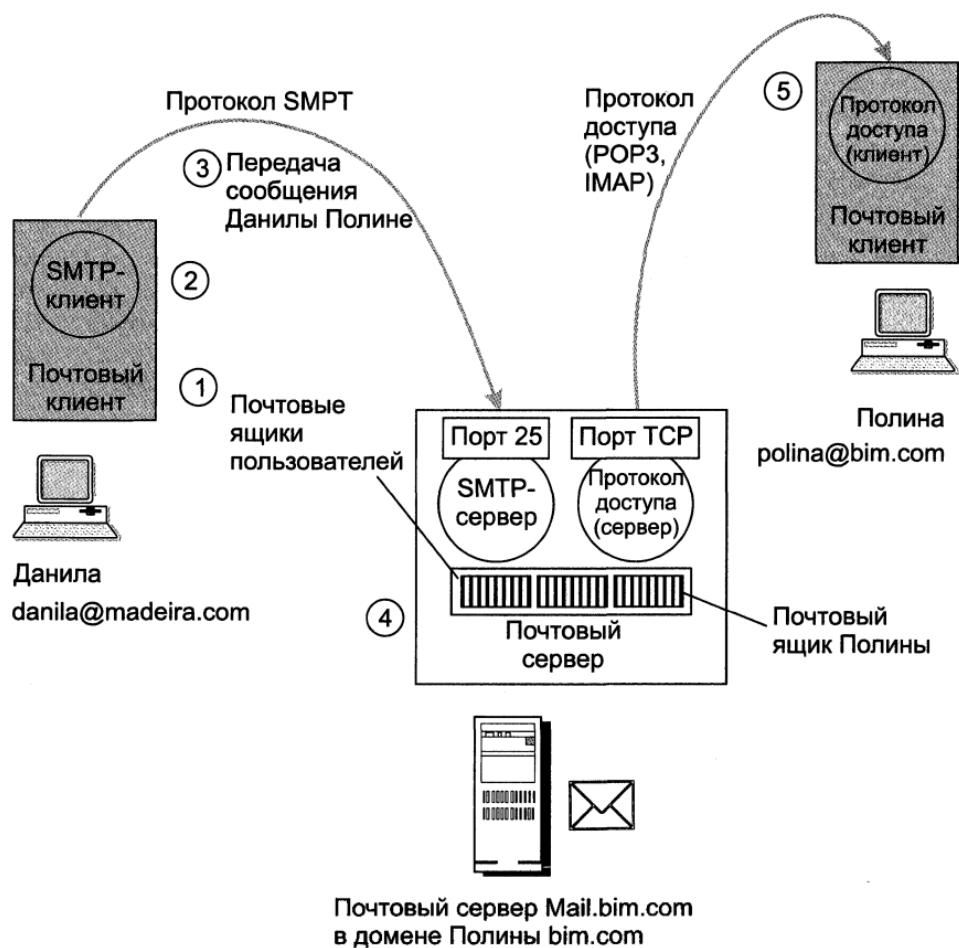


Рис. 5.2. Схема с выделенным почтовым сервером в принимающем домене

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`, 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 посыпает 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 буфера называют почтовыми ящиками. Важно заметить, что помимо Полины у почтового сервера имеется еще много других клиентов, и это усложняет его работу. То есть 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, наоборот, 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 (протокол POP3 ожидает запрос на установление TCP-соединения через порт 110, а 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.

Схема с двумя почтовыми серверами-посредниками

Прежде чем мы перейдем к сравнению двух протоколов доступа к почте, давайте посмотрим на еще одну схему организации почтовой службы, наиболее приближенную к реальности (рис. 5.3). Здесь передача сообщений между клиентами почты (на нашем 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 также 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.

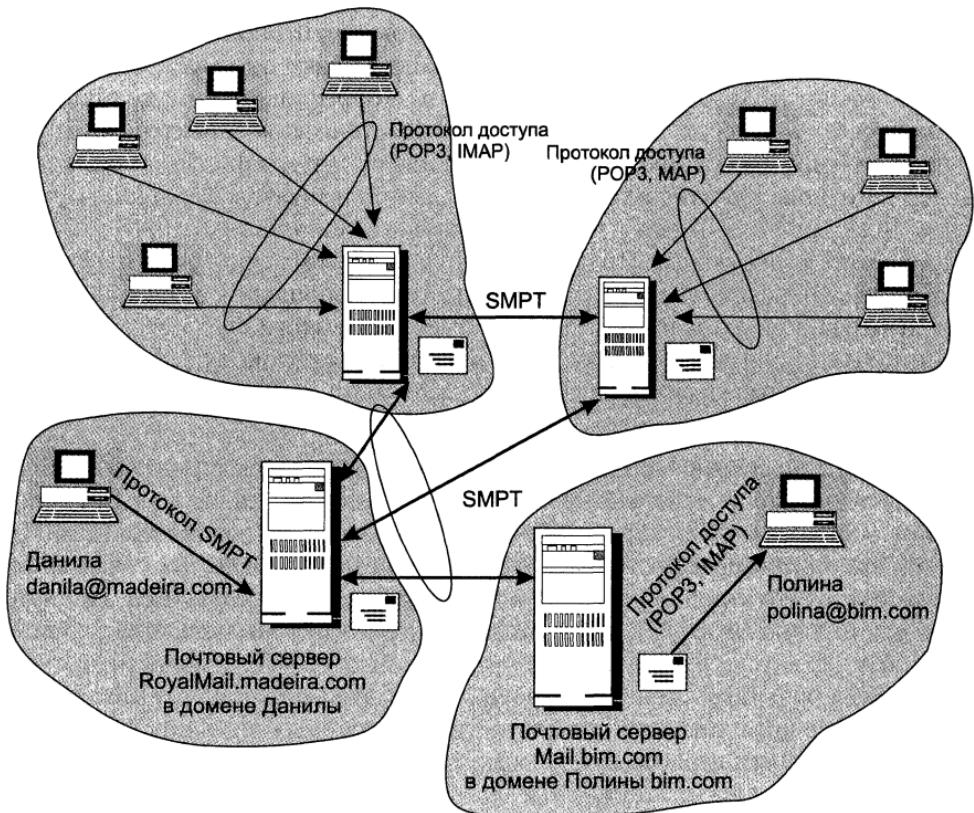


Рис. 5.3. Схема с выделенными почтовыми серверами в каждом домене

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.

Протоколы POP3 и IMAP

А теперь давайте, как мы и собирались, 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 этих 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 различных 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 будет «рассеяна» 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 являются столь критичными. Но и в этом случае проявляется еще один «дефект» этого протокола — клиент не может пропустить, не читая, ни одного письма, поступающего от сервера. То есть объемное и, возможно, совсем ненужное 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 ни обратилась бы 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.

Веб-служба

Изобретение службы World Wide Web (WWW), или Всемирной паутины, стоит в одном ряду с изобретениями телефона, радио и телевидения. Благодаря WWW люди получили возможность получать доступ 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 веб-служба. Заметим, что WWW 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 сделать без особых организационных забот и практически бесплатно.

Мы 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-или GIF-изображения, другие HTML-файлы, аудио- или видеофайлы.

HTML-страница, или HTML-файл, или **гипертекстовая страница**, содержит текст, написанный на языке HTML (HyperText Markup Language – язык разметки гипертекста). История появления этого языка связана с попытками программистов разработать средство, которое позволяло бы им 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, а важных выводов — курсивным или полужирным начертанием, 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 >** (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.

Существует особый тип тега, который имеет вид **< a href='...' ... >** и называется **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 может 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

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.

1. *Тип протокола доступа.* Помимо HTTP, здесь могут быть указаны и другие протоколы, такие как FTP, telnet, также позволяющие осуществлять удаленный доступ к файлам или компьютерам¹. Тем не менее основным протоколом доступа к веб-страницам является HTTP (как в нашем примере), и мы поговорим о нем немного позже.
2. *DNS-имя сервера.* Имя сервера, на котором хранится нужная страница. В нашем случае — это имя сайта www.olifer.co.uk.
3. *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

Как мы уже отмечали, 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**, или **агентом пользователя** веб-службы, представляет собой приложение, которое устанавливается на компьютере конечного пользователя и одной из важных функций которого является поддержание графического пользовательского интерфейса. Через этот интерфейс пользователь получает доступ к широкому набору услуг, главной из которых, конечно, является «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

¹ URL-адреса с самого начала предназначались не только для веб-служб, но и для других сервисов доступа к информации через Интернет.

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 конфигурированием самого браузера.

К числу наиболее популярных сейчас браузеров можно отнести Microsoft Internet Explorer, Mozilla Firefox компании Mozilla и последнее предложение компании Google — Chrome. Веб-браузер — это не единственный вид клиента, который может обращаться к веб-серверу. Этую роль могут исполнять любые программы и устройства, поддерживающие протокол HTTP, а также многие модели мобильных телефонов — для доступа в этом случае применяется специальный протокол WAP (Wireless Application Protocol — протокол беспроводных приложений).

Значительную часть своих функций браузер выполняет в тесной кооперации с веб-сервером. Как уже было сказано, клиент и сервер веб-службы связываются через сеть по протоколу HTTP. Это означает, что в клиентской части веб-службы присутствует клиентская часть HTTP, а в серверной — серверная часть 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 и Microsoft Internet Information Server.

Как и любой другой сервер, веб-сервер должен быть постоянно в активном состоянии, прослушивая TCP-порт 80, который является назначенным портом протокола HTTP. Как только сервер получает запрос от клиента, он устанавливает TCP-соединение и получает от клиента имя объекта, например, в виде /books/books.htm, после чего находит в своем каталоге этот файл, а также другие связанные с ним объекты и отсылает по TCP-соединению клиенту. Получив объекты от сервера, веб-браузер отображает их на экране (рис. 5.4). После отправки всех объектов страницы клиенту сервер разрывается с ним TCP-соединение. В дополнительные функции сервера входят также аутентификация клиента и проверка прав доступа данного клиента к данной странице.

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 её 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.

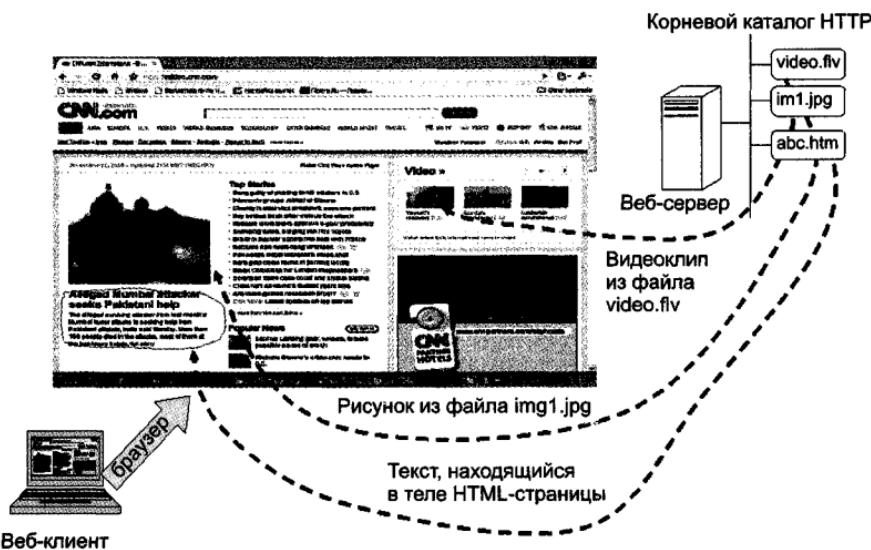


Рис. 5.4. Отображение веб-страницы

Протокол HTTP

HTTP (HyperText Transfer Protocol – протокол передачи гипертекста)¹ – это протокол прикладного уровня, во многом аналогичный протоколам FTP и SMTP. В настоящее время используются две версии протокола, HTTP/1.0 и HTTP/1.1

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. При этом TCP-соединения могут использоваться двумя разными способами:

- долговременное соединение** – передача в одном TCP-соединении нескольких объектов, причем время существования соединения определяется при конфигурировании веб-службы;
- кратковременное соединение** – передача в течение одного TCP-соединения только одного объекта.

Долговременное соединение, в свою очередь, может быть использовано двумя способами:

- последовательная передача запросов с паузами** – новый запрос посыпается только после получения ответа;
- Конвейерная передача** – это более эффективный способ, в котором следующий запрос посыпается до прибытия ответа на один или несколько

¹ RFC 1945, 2616.

предыдущих запросов (напоминает метод скользящего окна). Обычно по умолчанию степень параллелизма устанавливается на уровне 5–10, но у пользователя имеется возможность изменять этот параметр при конфигурировании клиента.

В HTTP 1.1 по умолчанию применяются постоянные соединения и конвейерный режим.

Формат HTTP-сообщений

В протоколе HTTP все сообщения состоят из текстовых строк. Сообщения и запросов, и ответов имеют единую обобщенную структуру из трех частей: обязательной стартовой строки, а также необязательных заголовков и тела сообщения. В табл. 5.1 приведены форматы и примеры стартовых строк и заголовков для запросов и ответов.

Как видно из таблицы, запросы и ответы имеют разные форматы стартовой строки. Каждая из них состоит из трех элементов, включающих поле *версии протокола HTTP*. В примере и в запросе, и в ответе указана версия HTTP/1.1. 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. Помимо этого метода в запросах протокол предусматривает и другие методы, такие как POST, который используется клиентом, например, для отправки электронной почты или в поисковых машинах, когда клиент запрашивает у сервера не определенный объект, а объекты, содержащие ключевые слова, помещенные в теле сообщения. Еще одним элементом стартовой строки является *URL-ссылка* на запрашиваемый объект — здесь это имя файла /books/books.htm.

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 для пользователя.

В настоящее время стандарты определяют пять классов кодов состояния:

- 1xx — информация о процессе передачи;
- 2xx — информация об успешном принятии и обработке запроса клиента (в таблице в примере стартовой строки ответа приведен код и соответствующая фраза 200 OK, сообщающие клиенту, что его запрос успешно обработан);
- 3xx — информация о том, что для успешного выполнения операции нужно произвести следующий запрос по другому URL-адресу, указанному в дополнительном заголовке *Location*;
- 4xx — 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);

- 5xx – 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).

Таблица 5.1. 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/1.0 200 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 размещен текст запрашиваемой страницы

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 авторизации¹ 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.

¹ Об аутентификации и авторизации читайте в последних разделах этой главы.

Динамические 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 было бы очень 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. Например, многие интернет-магазины поддерживают базу данных продаваемых товаров, и 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 для этого используются программные языки сценариев, такие как Perl, PHP или 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.

В том случае, когда программа работает на стороне клиента, 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. Примером может служить код, написанный на языке ActionScript, который иногда используется для программирования интерактивной анимации в играх. Однако для этого требуется еще один механизм, поддерживаемый современными браузерами, — механизм надстроек (add-on). Механизм надстроек является программным интерфейсом между браузером и внешними программами, которые расширяют функциональные возможности браузеров. Программа-надстройка обрабатывает объекты веб-страницы определенного типа, в данном случае — код ActionScript. Программой-надстройкой, которая понимает ActionScript, является Flash-плейер компании Adobe. Если Flash-плейер загружен в браузер, то динамическая веб-страница, в которой есть код ActionScript, будет правильно работать и создаст интерактивную анимацию. Другим популярным языком программирования страниц на стороне клиента является JavaScript.

При программировании содержания страницы на стороне сервера процесс выглядит немного сложнее, так как программный код страницы создает содержание 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).

Протокол передачи файлов

До появления службы WWW сетевая файловая служба на основе протокола **FTP** (File Transfer Protocol), описанная в спецификации RFC 959, долгое время была самой популярной службой доступа к удаленным данным в Интернете и корпоративных IP-сетях. FTP-серверы и FTP-клиенты имеются практически в каждой ОС, кроме того, для доступа ко всем еще популярным FTP-архивам используются FTP-клиенты, встроенные в браузеры.

Протокол FTP позволяет целиком переместить файл с удаленного компьютера на локальный и наоборот. FTP также поддерживает несколько команд просмотра удаленного каталога и перемещения по каталогам удаленной файловой системы. Поэтому FTP особенно удобно использовать для доступа к тем файлам, данные которых нет смысла просматривать удаленно, а гораздо эффективней целиком переместить на клиентский компьютер (например, файлы исполняемых модулей приложений).

В протокол FTP встроены примитивные средства авторизации удаленных пользователей на основе передачи по сети пароля в открытом виде. Кроме того, поддерживается анонимный доступ, не требующий указания имени пользователя и пароля, который часто рассматривается как более безопасный, так как не подвергает пароли пользователей угрозе перехвата.

Основные модули службы FTP

FTP-клиент состоит из трех основных функциональных модулей.

□ **User Interface** (аналог агента пользователя) — пользовательский интерфейс, принимающий от пользователя команды и отображающий состояние FTP-сеанса на экране. Пользовательский интерфейс зависит от программной реализации FTP-клиента. Наряду с традиционными клиентами, работающими в символьном режиме, имеются и графические оболочки, не требующие от пользователя знания символьных команд. Символьные клиенты обычно поддерживают следующий основной набор команд:

- **open имя_хоста** — открытие сеанса с удаленным сервером;
- **bye** — завершение сеанса с удаленным хостом и завершение работы утилиты ftp;

- `close` – завершение сеанса с удаленным хостом, утилита `ftp` продолжает работать;
 - `ls (dir)` – печать содержимого текущего удаленного каталога;
 - `get имя_файла` – копирование удаленного файла на локальный хост;
 - `put имя_файла` – копирование удаленного файла на удаленный сервер.
- User-PI – интерпретатор команд пользователя. Этот модуль взаимодействует с модулем Server-PI FTP-сервера.
- User-DTP – модуль, осуществляющий передачу данных файла по командам, получаемым от модуля User-PI по протоколу клиент-сервер. Этот модуль взаимодействует с локальной файловой системой клиента.
- FTP-сервер включает два модуля.
- Server-PI – модуль, который принимает и интерпретирует команды, передаваемые по сети модулем User-PI.
- Server-DTP – модуль, управляющий передачей данных файла по командам от модуля Server-PI. Взаимодействует с локальной файловой системой сервера.

Управляющий сеанс и сеанс передачи данных

FTP-клиент и FTP-сервер поддерживают параллельно два сеанса – управляющий сеанс и сеанс передачи данных. *Управляющий сеанс* открывается при установлении первоначального FTP-соединения клиента с сервером, причем в течение одного управляющего сеанса может последовательно выполняться несколько *сеансов передачи данных*, в рамках которых передаются или принимаются несколько файлов.

Общая схема взаимодействия клиента и сервера выглядит следующим образом.

1. FTP-сервер всегда открывает управляющий TCP-порт 21 для прослушивания, ожидая приход запроса на установление управляющего FTP-соединения от удаленного клиента.
2. После установления управляющего соединения FTP-клиент отправляет на сервер команды, которые уточняют параметры соединения: имя и пароль клиента; роль участников соединения (активная или пассивная); порт передачи данных; тип передачи; тип передаваемых данных (двоичные данные или код ASCII); директивы на выполнение действий (читать файл, писать файл, удалить файл и т. п.).
3. После согласования параметров пассивный участник соединения переходит в режим ожидания открытия соединения на порт передачи данных. Активный участник инициирует это соединение и начинает передачу данных.

4. После окончания передачи данных соединение по портам данных закрывается, а управляющее соединение остается открытым. Пользователь может по управляющему соединению активизировать новый сеанс передачи данных.

Порты передачи данных выбирает FTP-клиент (по умолчанию клиент может использовать для передачи данных порт управляющего сеанса), а сервер должен задействовать порт, номер которого на единицу меньше номера порта клиента.

Команды для взаимодействия FTP-клиента с FTP-сервером

В протоколе FTP предусматриваются специальные команды для взаимодействия FTP-клиента с FTP-сервером (не следует их путать с командами пользовательского интерфейса клиента, ориентированными на применение человеком). Эти команды делятся на три группы.

1. *Команды управления доступом к системе* доставляют серверу имя и пароль клиента, изменяют текущий каталог на сервере, повторно инициализируют, а также завершают управляющий сеанс.
2. *Команды управления потоком данных* устанавливают параметры передачи данных. Служба FTP может применяться для передачи разных типов данных (код ASCII или двоичные данные), работать как со структурированными данными (файл, запись, страница), так и с неструктурными.
3. *Команды службы FTP* управляют передачей файлов, операциями над удаленными файлами и каталогами. Например, команды RETR и STOR запрашивают передачу файла, соответственно, от сервера на клиентский хост и наоборот. Параметрами каждой из этих команд является имя файла. Может быть задано также смещение от начала файла — это позволяет начать передачу файла с определенного места при непредвиденном разрыве соединения. Команды DELE, MKD, RMD, LIST, соответственно, удаляют файл, создают каталог, удаляют каталог и передают список файлов текущего каталога. Каждая команда протокола FTP передается в виде одной строки кода ASCII.

Системы управления сетью и протокол SNMP

Система управления сетью (Network Management System, NMS) — это совокупность программных средств, ориентированных на управление коммуникационным оборудованием и контроль сетевого трафика.

Обычно система управления работает в *автоматизированном* режиме, выполняя наиболее простые действия по управлению сетью автоматически, а сложные решения предоставляя принимать человеку на основе подготовленной системой информации.

Схема «менеджер — агент — управляемый объект»

Основным элементом любой системы управления сетью является схема взаимодействия **«менеджер — агент — управляемый объект»** (рис. 5.5). На основе этой схемы могут быть построены системы практически любой сложности с большим количеством агентов, менеджеров и ресурсов разного типа.

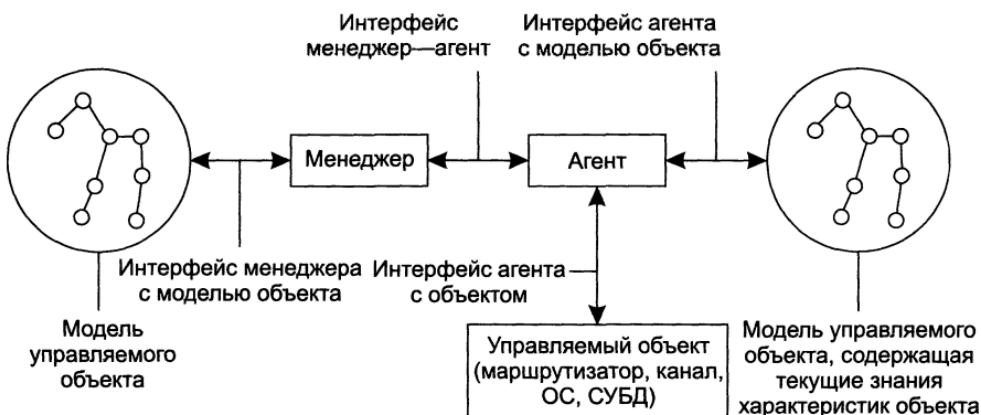


Рис. 5.5. Схема взаимодействия «менеджер — агент — управляемый объект»

Чтобы можно было автоматизировать управление объектами сети, создается некоторая **модель управляемого объекта**, называемая **базой данных управляющей информации** (Management Information Base, MIB). MIB отражает только те характеристики объекта, которые нужны для его контроля. Например, модель маршрутизатора обычно включает такие характеристики, как количество портов, их тип, таблица маршрутизации, количество кадров и пакетов протоколов канального, сетевого и транспортного уровней, прошедших через эти порты.

Менеджер и агент работают с одной и той же моделью управляемого объекта, однако в использовании этой модели агентом и менеджером имеются существенные различия. Агент наполняет MIB управляемого объекта текущими значениями его характеристик, а менеджер извлекает из MIB данные, на основании которых он узнает, какие характеристики он может запросить у агента и какими параметрами объекта можно управлять. Таким образом, агент является посредником между управляемым объектом и менеджером. Агент поставляет менеджеру только те данные, которые предусматриваются MIB.

Протокол SNMP

Менеджер и агент взаимодействуют по стандартному протоколу, в качестве которого часто используется так называемый **простой протокол управления сетью** (Simple Network Management Protocol, SNMP). Этот протокол позволяет менеджеру запрашивать значения параметров, хранящихся в MIB, а также передавать агенту информацию, на основе которой тот должен управлять объектом. Особенностью протокола является его чрезвычайная простота — он включает в себя всего несколько команд.

- **Get-request** — используется менеджером для получения от агента значения какого-либо объекта по его имени.
- **GetNext-request** — позволяет менеджеру извлечь значение следующего объекта (без указания его имени) при последовательном просмотре таблицы объектов.
- **Get-response** — с помощью этой команды SNMP-агент передает менеджеру ответ на команду **Get-request** или **GetNext-request**.
- **Set** — позволяет менеджеру изменять значения какого-либо объекта. С помощью команды **Set** и происходит собственно управление устройством. Агент должен «понимать» смысл значений объекта, который используется для управления устройством, и на основании этих значений выполнять реальное управляющее воздействие — отключить порт, приписать порт определенной линии VLAN и т. п. Команда **Set** пригодна также для задания условия, при выполнении которого SNMP-агент должен послать менеджеру соответствующее сообщение. Может быть определена реакция на такие события, как инициализация агента, рестарт агента, обрыв связи, восстановление связи, неверная аутентификация, потеря ближайшего маршрутизатора. Если происходит любое из этих событий, то агент инициализирует прерывание.
- **Trap** — используется агентом для сообщения менеджеру о возникновении особой ситуации.

Протокол SNMP для передачи данных между агентами и менеджерами использует дейтаграммный транспортный протокол UDP, не обеспечивающий надежной доставки сообщений, однако менее загружающий управляемые устройства, чем более надежный протокол TCP.

Структура систем управления

Обычно менеджер работает на отдельном компьютере, взаимодействуя с несколькими агентами.

Агенты могут встраиваться в управляемое оборудование, а могут и работать на отдельном компьютере, связанном с управляемым оборудованием. Для получения требуемых данных об объекте, а также для выдачи на него управляющих воздействий агент должен иметь возможность взаимодействовать с ним.

Однако многообразие типов управляемых объектов не позволяет стандартизировать способ взаимодействия агента с объектом. Эта задача решается разработчиками при встраивании агентов в коммуникационное оборудование или в операционную систему. Агент может снабжаться специальными датчиками для получения информации, например датчиками релейных контактов или датчиками температуры. Агенты могут различаться разным уровнем интеллекта, обладая как самым минимальным интеллектом, необходимым для подсчета проходящих через оборудование кадров и пакетов, так и весьма высоким, достаточным для самостоятельных действий по выполнению последовательности управляющих команд в аварийных ситуациях, построению временных зависимостей, фильтрации аварийных сообщений и т. п.

Схема «менеджер – агент – управляемый объект» позволяет строить достаточно сложные в структурном отношении распределенные системы управления (рис. 5.6).

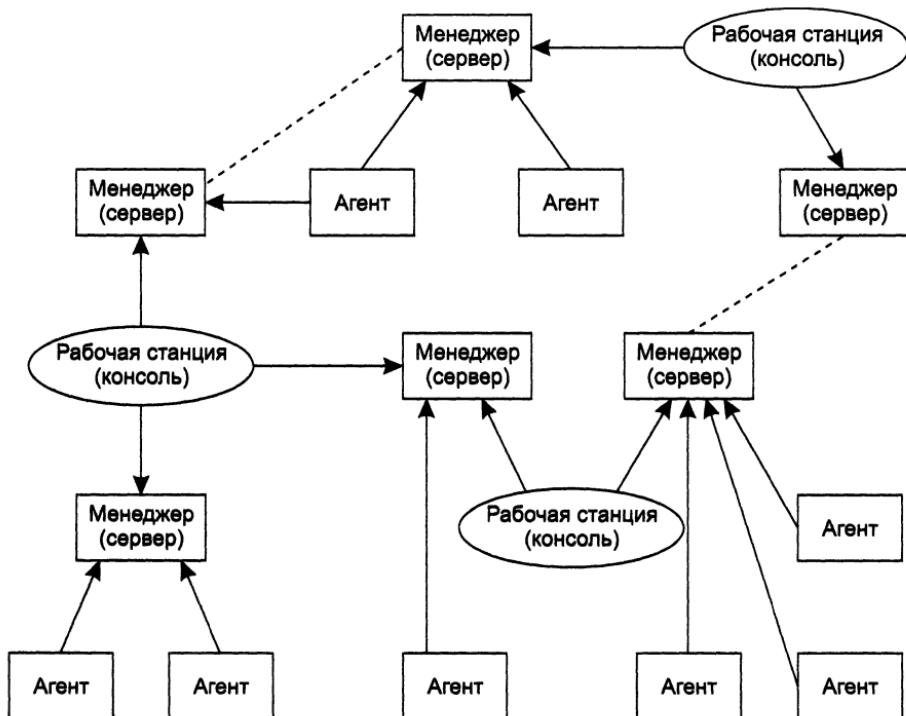


Рис. 5.6. Структуры систем управления

Как показано на рисунке, каждый агент управляет определенным элементом сети, параметры которого помещает в соответствующую базу MIB. Менеджеры извлекают данные из MIB 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.

Протокол telnet

Режим удаленного управления поддерживается специальным протоколом прикладного уровня, работающим поверх протоколов, реализующих транспортное соединение удаленного узла 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, который работает в архитектуре клиент-сервер, обеспечивает эмуляцию алфавитно-цифрового терминала, ограничивая пользователя режимом командной строки.

При нажатии клавиши соответствующий код перехватывается клиентом telnet, помещается в TCP-сообщение и отправляется через сеть удаленному узлу, которым пользователь пытается управлять. При поступлении на узел назначения код нажатой клавиши извлекается из TCP-сообщения сервером telnet и передается операционной системе узла. ОС рассматривает сеанс telnet как один из сеансов локального для нее пользователя, а коды команд, поступающие из сети, как коды, генерируемые нажатием клавиш. Все ответные сообщения операционной системы при таком режиме работы упаковываются сервером telnet в TCP-сообщения и по сети отправляются клиенту telnet. Клиент telnet извлекает символные сообщения удаленной ОС и отображает их 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. Сегодня этот протокол в публичном домене Интернета используется редко, так как мало желающих предоставлять посторонним лицам возможность управлять собственным компьютером. Хотя для защиты от несанкционированного доступа в технологии telnet применяются пароли, они передаются через сеть в виде обычного текста, поэтому могут быть легко перехвачены и применены. Из-за этого telnet преимущественно используется в пределах одной локальной сети, где возможностей для перехвата пароля гораздо меньше. Сегодня основной областью применения telnet является управление не компьютерами, а коммуникационными устройствами — маршрутизаторами, коммутаторами и хабами. Таким образом, он уже скорее не пользовательский протокол, а протокол администрирования, то есть альтернатива SNMP.

Тем не менее различие между протоколами telnet и SNMP принципиальное. Telnet предусматривает обязательное участие человека в процессе администрирования, так как, по сути, протокол только передает команды, которые

вводит администратор при конфигурировании или мониторинге маршрутизатора или другого 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, создаваемой 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.

Удаленное управление имеет свои достоинства и недостатки. Для пользователя часто удобно действовать более мощный компьютер, установленный в сети предприятия, а не свой домашний. Кроме того, получив терминальный доступ, он может запустить на удаленном компьютере любое приложение, а не только службу WWW или FTP. Удаленное управление также очень экономично потребляет пропускную способность сети, особенно при эмуляции режима командной строки. Действительно, в этом случае по сети передаются только коды клавиш и экранные символы, а не файлы или страницы веб-документов.

Недостаток удаленного управления состоит в его опасности для сети предприятия при несанкционированном доступе. Кроме того, администратору трудно контролировать потребление ресурсов компьютера, находящегося под удаленным управлением.

В тех случаях, когда степень защиты, предлагаемая протоколом telnet, не удовлетворяет пользователей, применяется более защищенный протокол удаленного управления SSH (Secure SHell).

Службы сетевой безопасности

Безопасность компьютера и сетевая безопасность

При рассмотрении темы безопасности информационных систем обычно выделяют две группы проблем: безопасность компьютера и сетевая безопасность. К *безопасности компьютера* относят все проблемы защиты данных, хранящихся и обрабатывающихся компьютером, который рассматривается как автономная система. Эти проблемы решаются средствами операционных систем и приложений, таких как базы данных, а также встроенными аппаратными средствами компьютера. Под *сетевой безопасностью* понимают все вопросы, связанные с взаимодействием устройств в сети — это, прежде всего, защита данных в момент их передачи по линиям связи и защита от несанкционированного удаленного доступа в сеть. И хотя подчас проблемы компьютерной и сетевой безопасности трудно отделить друг от друга, настолько тесно они связаны, совершенно очевидно, что сетевая безопасность имеет свою специфику.

Автономно работающий компьютер можно эффективно защитить от внешних покушений разнообразными способами, например, просто запереть на замок клавиатуру или снять жесткий накопитель и поместить его в сейф. Компьютер, работающий в сети, по определению не может полностью отгородиться от мира, он должен общаться с другими компьютерами, возможно удаленными от него на большое расстояние, поэтому обеспечение безопасности в сети является задачей значительно более сложной. Логический вход чужого пользователя в ваш компьютер является штатной ситуацией, если вы работаете в сети. Обеспечение безопасности в такой ситуации сводится к тому, чтобы сделать это проникновение контролируемым — каждому пользователю сети должны быть четко определены его права по доступу к информации, внешним устройствам и выполнению системных действий на каждом из компьютеров сети.

Помимо проблем, порождаемых возможностью удаленного входа в сетевые компьютеры, сети по своей природе подвержены еще одному виду опасности — перехвату и анализу сообщений, передаваемых по сети, а также созданию «ложного» трафика. Большая часть средств обеспечения сетевой безопасности направлена на предотвращение именно этого типа нарушений.

Вопросы сетевой безопасности приобретают особое значение сейчас, когда при построении корпоративных сетей наблюдается переход от выделенных каналов к публичным сетям (Интернет, сеть провайдера). Поставщики услуг публичных сетей пока редко обеспечивают защиту пользовательских данных при их транспортировке по своим магистралям, возлагая на пользователей заботы по их конфиденциальности, целостности и доступности.

Конфиденциальность, целостность и доступность данных

Безопасная информационная система — это система, которая, во-первых, защищает данные от несанкционированного доступа, во-вторых, всегда готова предоставить их своим пользователям, в-третьих, надежно хранит информацию и гарантирует неизменность данных. Другими словами, безопасная система по определению обладает свойствами конфиденциальности, доступности и целостности.

- **Конфиденциальность (confidentiality)** — это гарантия того, что секретные данные будут доступны только тем пользователям, которым этот доступ разрешен (такие пользователи называются авторизованными).
- **Доступность (availability)** — это гарантия того, что авторизованные пользователи всегда получат доступ к данным.
- **Целостность (integrity)** — это гарантия сохранности данными правильных значений, которая обеспечивается запретом для неавторизованных пользователей каким-либо образом изменять, модифицировать, разрушать или создавать данные.

Требования безопасности могут меняться в зависимости от назначения системы, характера используемых данных и типа возможных угроз. Трудно представить систему, для которой были бы не важны свойства целостности и доступности, но свойство 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 в данном случае не требуется. Однако требования целостности и доступности остаются актуальными.

Действительно, если вы не предпримете специальных мер по обеспечению целостности данных, злоумышленник может изменить данные на вашем сервере и нанести этим ущерб вашему предприятию. Преступник может, например, внести такие изменения в помещенный на веб-сервере прайс-лист, которые снизят конкурентоспособность вашего предприятия, или испортить коды свободно распространяемого вашей фирмой программного продукта, что, безусловно, скажется на ее деловом имидже.

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 IP-пакетами с неправильным обратным адресом, которые в соответствии с логикой работы этого протокола могут вызывать тайм-ауты и в конечном счете сделать сервер недоступным для всех остальных запросов.

Понятия конфиденциальности, доступности и целостности могут быть определены по отношению не только к информации, но и к другим ресурсам вычислительной сети, например, внешним *устройствам* или *приложениям*. Так, неограниченный доступ к устройству печати позволяет злоумышленнику получать копии распечатываемых документов и изменять параметры, что может привести к изменению очередности работ и даже к выводу устройства из строя. Свойство конфиденциальности, примененное к устройству печати, можно интерпретировать так, что доступ к устройству имеют те и только те пользователи, которым этот доступ разрешен, причем они могут выполнять только те операции с устройством, которые для них определены. Свойство доступности устройства означает его готовность к использованию всякий раз, когда в этом возникает необходимость. А свойство целостности может быть определено как свойство неизменности параметров данного устройства. Легальность применения сетевых устройств важна не только постольку, поскольку она влияет на безопасность данных. Устройства могут предоставлять различные услуги, такие как распечатка текстов, отправка факсов, доступ в Интернет, электронная почта и т. п., незаконное потребление которых, наносящее материальный ущерб предприятию, также является нарушением безопасности системы.

Угрозы, атаки, риски

Любое действие, которое направлено на нарушение конфиденциальности, целостности и (или) доступности информации, а также на нелегальное использование других ресурсов сети, называется **угрозой**.

Реализованная угроза называется **атакой**.

Риск – это вероятностная оценка величины возможного ущерба, который может понести владелец информационного ресурса в результате успешно проведенной атаки. Значение риска тем выше, чем более уязвимой является существующая система безопасности и чем выше вероятность реализации атаки.

Угрозы могут быть разделены на умышленные и неумышленные. *Неумышленные* угрозы вызываются ошибочными неквалифицированными действиями лояльных сотрудников, а также являются следствием ненадежной работы программных и аппаратных средств системы. Так, из-за отказа диска, контроллера диска или всего файлового сервера могут оказаться недоступными данные, критически важные для работы предприятия. Умышленные угрозы могут либо ограничиваться пассивным чтением данных или мониторингом системы, либо включать в себя активные действия, например, незаконное проникновение в один из компьютеров сети под видом легального пользователя, разрушение системы с помощью программ-вирусов или «подслушивание» внутрисетевого трафика.

Одним из способов незаконного проникновения в сеть является применение «чужих» *паролей*, полученных путем подглядывания, расшифровки файла паролей, подбора паролей или получения пароля путем анализа сетевого трафика. Особенно опасно проникновение злоумышленника под именем пользователя, наделенного большими полномочиями, например, администратора сети. Такого рода угрозы могут исходить и от легальных пользователей сети, которые в рамках своих полномочий пытаются выполнять действия, выходящие за рамки их должностных обязанностей. Существующая статистика говорит о том, что едва ли не половина всех попыток нарушения безопасности системы исходит от сотрудников предприятия, которые как раз и являются легальными пользователями сети.

Подбор паролей злоумышленник выполняет с помощью специальных программ, которые работают путем перебора слов из некоторого файла, содержащего большое количество слов. Содержимое файла-словаря формируется с учетом психологических особенностей человека, которые выражаются в том, что человек выбирает в качестве пароля легко запоминаемые слова или буквенные сочетания.

Еще один способ получения пароля – внедрение в чужой компьютер **троянского коня**. Так называют резидентную программу, работающую без ведома

хозяина данного компьютера и выполняющую действия, заданные злоумышленником. В частности, такого рода программа может считывать коды пароля, вводимого пользователем во время логического входа в систему.

Программа троянского коня всегда маскируется под какую-нибудь полезную утилиту или игру, а производит действия, разрушающие систему. По такому принципу действуют и **программы-вирусы**, отличительной особенностью которых является способность «заражать» другие файлы, внедряя в них собственные копии. Чаще всего вирусы поражают исполняемые файлы. Когда такой исполняемый код загружается в оперативную память для выполнения, вместе с ним получает возможность исполнить свои вредительские действия вирус. Вирусы могут привести к повреждению или даже полной утрате информации.

«Подслушивание» *внутрисетевого трафика* — это незаконный мониторинг сети, захват и анализ сетевых сообщений. Существует много доступных программных и аппаратных анализаторов трафика, которые делают эту задачу достаточно тривиальной. Использование общественных сетей (речь в основном идет об Интернете) еще более усугубляет ситуацию. Действительно, работа в Интернете добавляет к опасности перехвата данных, передаваемых по линиям связи, опасность несанкционированного входа в узлы сети, поскольку наличие огромного числа хакеров в Интернете повышает вероятность попыток незаконного проникновения в компьютер. Это представляет постоянную угрозу для сетей, подсоединенных к Интернету.

Интернет сам является целью для разного рода злоумышленников. Поскольку Интернет создавался как открытая система, предназначенная для свободного обмена информацией, совсем не удивительно, что практически все протоколы стека TCP/IP имеют «врожденные» недостатки защиты. Эксплуатируя эти недостатки, злоумышленники все чаще предпринимают попытки несанкционированного доступа к информации, хранящейся на узлах Интернета.

Построение и поддержка безопасной системы требует системного подхода. В соответствии с этим подходом, прежде всего, необходимо осознать весь спектр возможных угроз для конкретной сети и для каждой из этих угроз продумать тактику ее отражения. В этой борьбе можно и нужно использовать самые разноплановые средства и приемы — морально-этические и законодательные, административные и психологические, защитные возможности программных и аппаратных средств сети.

Шифрование, сертификат, электронная подпись

В разных программных и аппаратных продуктах, предназначенных для защиты данных, часто используются одинаковые подходы, приемы и технические решения. К таким базовым технологиям безопасности относятся аутентификация, авторизация, аудит и технология защищенного канала.

Шифрование — это краеугольный камень всех служб информационной безопасности, будь то система аутентификации или авторизации, средства создания защищенного канала или способ безопасного хранения данных.

Любая процедура шифрования, превращающая информацию из обычного «понятного» вида в «нечитабельный» зашифрованный вид, естественно, должна быть дополнена процедурой дешифрования, которая, будучи примененной к зашифрованному тексту, снова делает его понятным. Пара процедур — шифрование и дешифрование — называется **крипtosистемой**.

Информацию, над которой выполняются функции шифрования и дешифрования, будем условно называть «текст», учитывая, что это может быть также числовой массив или графические данные.

В современных алгоритмах шифрования предусматривается наличие параметра — **секретного ключа**. В криптографии принято правило Керкхоффа: «Стойкость шифра должна определяться только секретностью ключа». Так, все стандартные алгоритмы шифрования (например, DES, PGP) широко известны, их детальное описание содержится в легко доступных документах, но от этого их эффективность не снижается. Злоумышленнику может быть все известно об алгоритме шифрования, кроме секретного ключа (следует отметить, однако, что существует немало фирменных алгоритмов, описание которых не публикуется).

Алгоритм шифрования считается *раскрытым*, если найдена процедура, позволяющая подобрать ключ за реальное время. Сложность алгоритма раскрытия является одной из важных характеристик крипtosистемы и называется **криптостойкостью**.

Существует два класса крипtosистем — 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) открытый ключ шифрования не совпадает с секретным ключом дешифрования.

На рис. 5.7 приведена классическая модель **симметричной крипtosистемы**. В данной модели три участника: отправитель, получатель, злоумышленник. Задача отправителя заключается в том, чтобы по открытому каналу передать некоторое сообщение в защищенном виде. Для этого он на ключе k зашифровывает открытый текст X и передает шифрованный текст Y . Задача получателя заключается в том, чтобы расшифровать Y и прочитать сообщение X . Предполагается, что отправитель имеет свой источник ключа. Генерированный ключ заранее по надежному каналу передается получателю. Задача злоумышленника заключается в перехвате и чтении передаваемых сообщений, а также в имитации ложных сообщений.

Модель является универсальной — если зашифрованные данные хранятся в компьютере и никуда не передаются, отправитель и получатель совмещаются в одном лице, а в роли злоумышленника выступает некто, имеющий доступ к компьютеру в ваше отсутствие.

Наиболее популярным стандартным симметричным алгоритмом шифрования данных является **DES** (Data Encryption Standard). Для того чтобы повысить криптостойкость алгоритма DES, иногда применяют его усиленный вариант, называемый «тройным алгоритмом DES», который включает троекратное шифрование с использованием двух разных ключей.

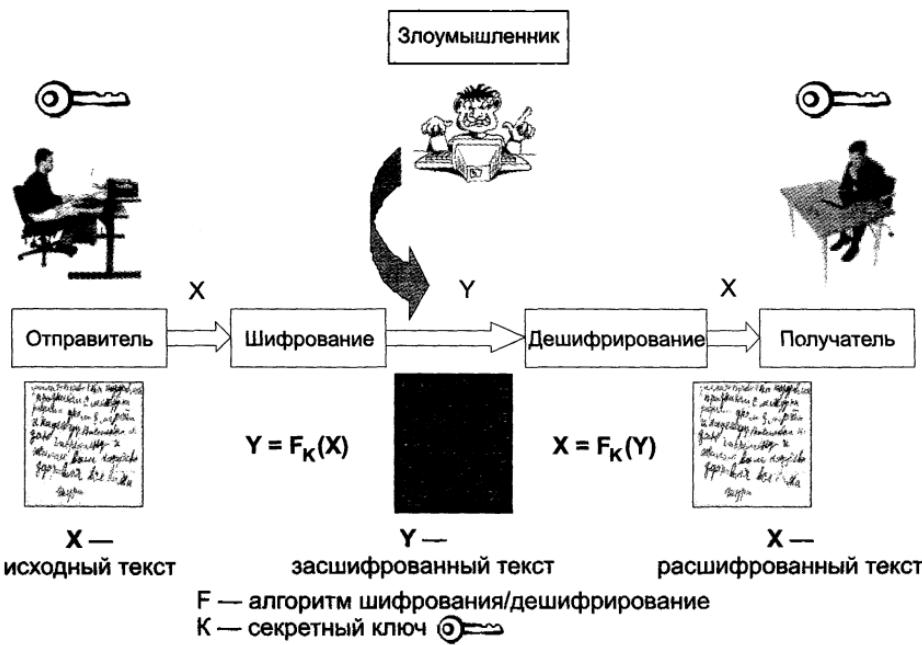


Рис. 5.7. Модель симметричного шифрования

В симметричных алгоритмах главную проблему представляют ключи. Во-первых, криптостойкость многих симметричных алгоритмов зависит от качества ключа, а это предъявляет повышенные требования к службе генерации ключей. Во-вторых, принципиальной является надежность канала передачи ключа второму участнику секретных переговоров. В системе с n абонентами, желающими обмениваться секретными данными по принципу «каждый с каждым», потребуется $n \times (n - 1)/2$ ключей, которые должны быть сгенерированы и распределены надежным образом. То есть количество ключей пропорционально квадрату количества абонентов, что при большом числе абонентов делает задачу чрезвычайно сложной. Несимметричные алгоритмы, основанные на использовании открытых ключей, снимают эту проблему.

В модели **криптосхемы с открытым ключом** также три участника: отправитель, получатель и злоумышленник (рис. 5.8). Задача отправителя заключается в том, чтобы по открытому каналу связи передать некоторое сообщение в защищенном виде. Получатель генерирует на своей стороне два ключа: открытый E и закрытый D . Закрытый ключ D (часто называемый также личным

ключом) абонент должен сохранять в защищенном месте, а открытый ключ E он может передать всем, с кем хочет поддерживать защищенные отношения. Открытый ключ используется для шифрования текста, но расшифровать текст можно только с помощью закрытого ключа. Поэтому открытый ключ передается отправителю в *незащищенном* виде. Отправитель, применяя открытый ключ получателя, шифрует сообщение X и передает его получателю. Получатель расшифровывает сообщение своим закрытым ключом D .

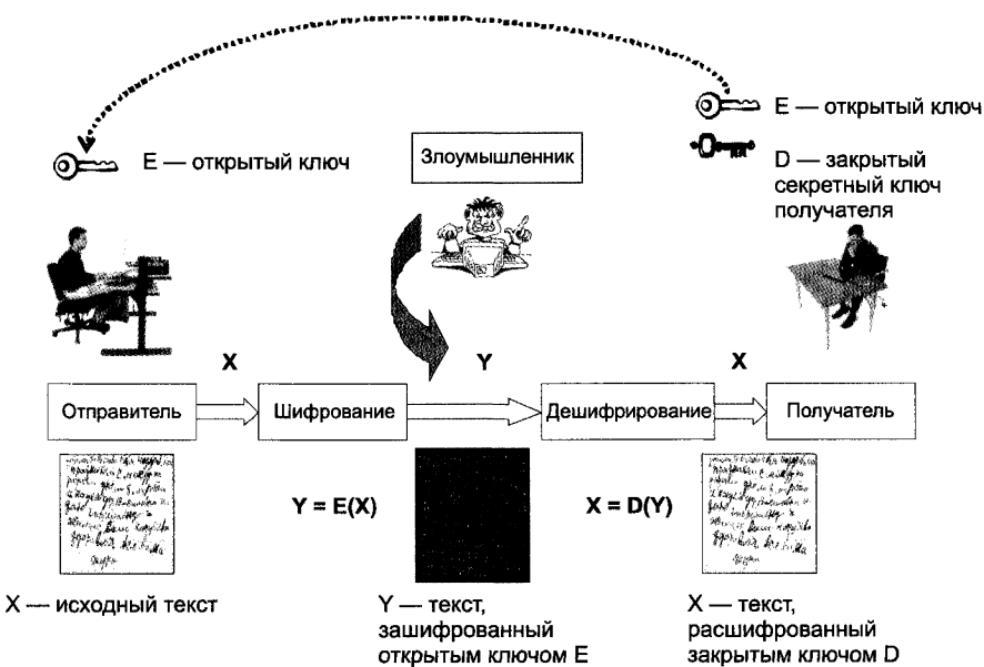


Рис. 5.8. Модель криптосхемы с открытым ключом

Очевидно, что числа, одно из которых используется для шифрования текста, а другое — для дешифрирования, не могут быть независимыми друг от друга, а значит, имеется принципиальная возможность вычисления закрытого ключа по открытому. Это действительно так, однако данные вычисления требуют огромного времени.

Поясним принципиальную связь между закрытым и открытым ключами следующей аналогией.

Пусть абонент 1 (рис. 5.9, а) решает вести секретную переписку со своими сотрудниками на малоизвестном языке, например, санскрите. Для этого он обзаводится санскритско-русским словарем, а всем своим абонентам посыпает русско-санскритские словари. Каждый из них, пользуясь словарем, пишет сообщения на санскрите и посыпает их абоненту 1, который переводит их на русский язык, пользуясь доступным только ему санскритско-русским словарем.

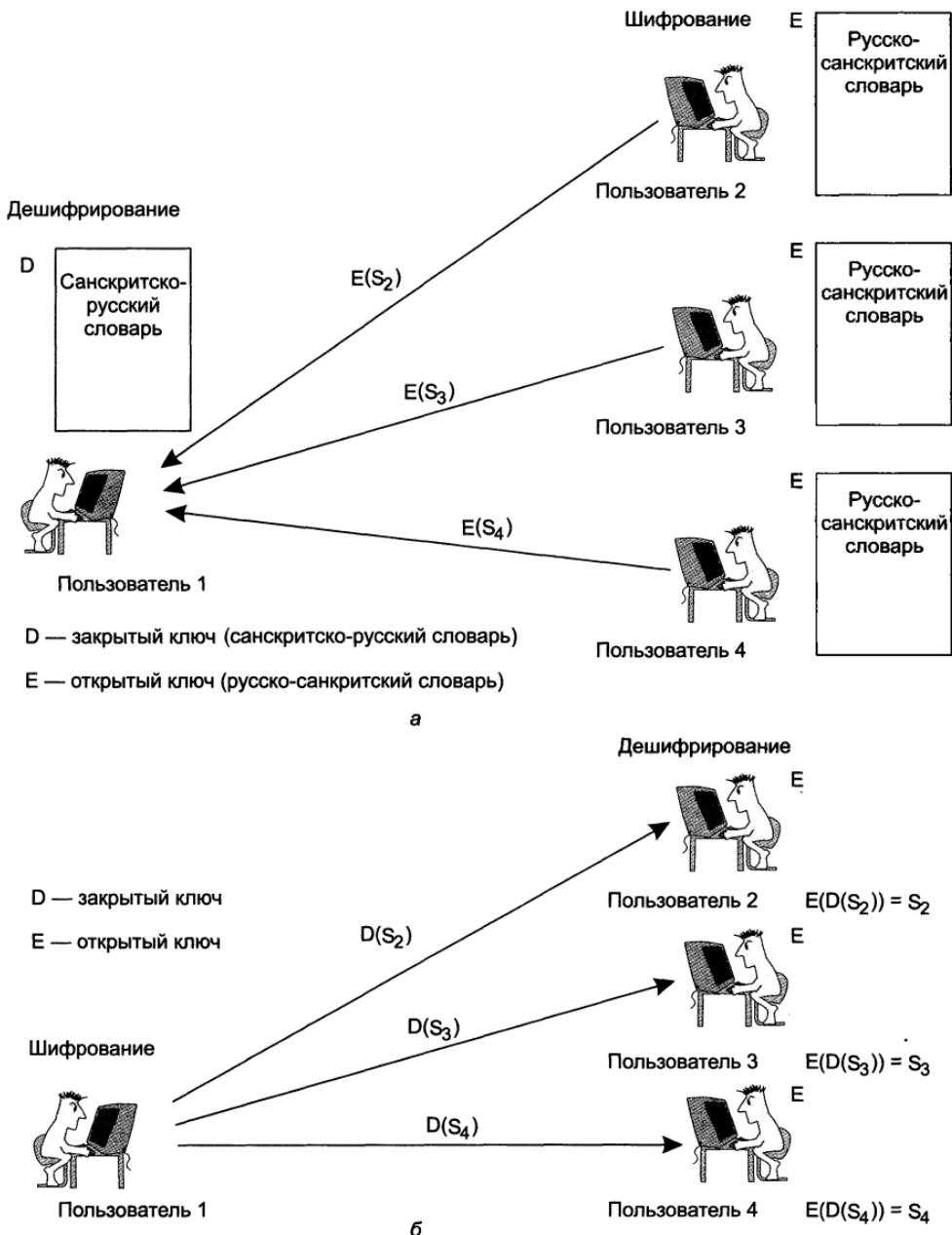


Рис. 5.9. Две схемы использования открытого и закрытого ключей

Очевидно, что здесь роль открытого ключа E играет русско-санскритский словарь, а роль закрытого ключа D — санскритско-русский словарь. Могут ли абоненты 2, 3 и 4 прочитать чужие сообщения S_2, S_3, S_4 , которые посылает

каждый из них абоненту 1? Вообще-то нет, так как для этого им нужен санскритско-русский словарь, обладателем которого является только абонент 1. Однако теоретическая возможность этого имеется, так как затратив массу времени, можно прямым перебором составить санскритско-русский словарь по русско-санскритскому. Такая процедура, требующая больших временных затрат, является отдаленной аналогией восстановления закрытого ключа по открытому.

На рис. 5.9, б показана другая схема использования открытого и закрытого ключей, целью которой является подтверждение авторства (аутентификация) посылаемого сообщения. В этом случае поток сообщений имеет обратное направление — от абонента 1, обладателя закрытого ключа D , к его корреспондентам, обладателям открытого ключа E . Если абонент 1 хочет аутентифицировать себя (поставить свою **электронную подпись**), то он шифрует известный текст своим закрытым ключом D и передает шифровку своим корреспондентам. Если им удаётся расшифровать текст открытым ключом абонента 1, это доказывает, что текст был зашифрован его же закрытым ключом, а значит, именно он является автором этого сообщения. Заметим, что в этом случае сообщения S_2 , S_3 , S_4 , адресованные разным абонентам, не являются секретными, так как все они — обладатели одного и того же открытого ключа, с помощью которого они могут расшифровывать все сообщения, поступающие от абонента 1.

Если же нужна взаимная аутентификация и двунаправленный секретный обмен сообщениями, то каждая из общающихся сторон генерирует собственную пару ключей и посылает открытый ключ своему корреспонденту.

Для того чтобы в сети все n абонентов имели возможность не только принимать зашифрованные сообщения, но и посыпать 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 дешифрирования. Таким образом решается проблема масштабируемости — квадратичная зависимость количества ключей от числа абонентов в симметричных алгоритмах заменяется линейной зависимостью в несимметричных алгоритмах. При этом задача секретной доставки ключа становится неактуальной. Злоумышленнику нет смысла стремиться завладеть открытым ключом, поскольку это не дает возможности расшифровывать текст или вычислить закрытый ключ.

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, посыпаемые легальному пользователю, и отправлять свои сообщения от его имени. Проще всего было бы распространять списки, связывающие имена пользователей с их открытыми ключами широковещательно путем публикаций в средствах массовой информации (бюллетени, специализированные журналы и т. п.). Однако

при таком подходе мы снова, как и в случае с паролями, сталкиваемся с плохой масштабируемостью. Решением этой проблемы является технология цифровых сертификатов. **Сертификат** в данном контексте — это электронный документ, который связывает конкретного пользователя с конкретным ключом.

В настоящее время одним из наиболее популярных криптоалгоритмов с открытым ключом является криптоалгоритм **RSA**, названный так по первым буквам фамилий его создателей (Rivest, Shamir, Adleman).

Во многих базовых технологиях безопасности используются **односторонние функции** (one-way function) шифрования, называемые также хэш- (hash function), или дайджест-функциями (digest function).

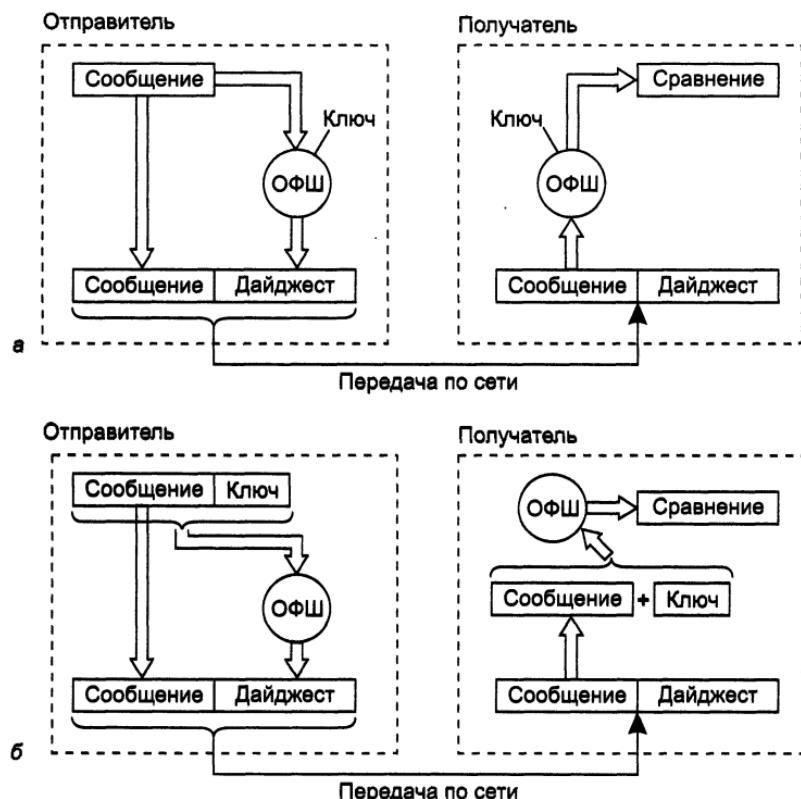


Рис. 5.10. Односторонние функции шифрования

Односторонняя функция, примененная к шифруемым данным, дает в результате значение (дайджест), состоящее из фиксированного небольшого количества байтов (рис. 5.10, а). Дайджест передается вместе с исходным сообщением. Получатель сообщения, зная, какая односторонняя функция

шифрования (ОФШ) была применена для получения дайджеста, заново вычисляет его, используя незашифрованную часть сообщения. Если значения полученного и вычисленного дайджестов совпадают, значит, содержимое сообщения не было подвергнуто никаким изменениям. Знание дайджеста не дает возможности восстановить исходное сообщение, но зато позволяет проверить целостность данных.

Дайджест является своего рода контрольной суммой для исходного сообщения. Однако имеется и существенное отличие. Использование контрольной суммы является средством проверки целостности передаваемых сообщений по ненадежным линиям связи. Это средство не направлено на борьбу со злоумышленниками, которым в такой ситуации ничто не мешает подменить сообщение, добавив к нему новое значение контрольной суммы. Получатель в таком случае не заметит никакой подмены. В отличие от контрольной суммы, при вычислении дайджеста требуются секретные ключи. В случае если для получения дайджеста применялась односторонняя функция с параметром, который известен только отправителю и получателю, любая модификация исходного сообщения будет немедленно обнаружена.

Наиболее популярной в системах безопасности в настоящее время является серия хэш-функций MD2, MD4, MD5. Все они генерируют дайджесты фиксированной длины 16 байт.

Идентификация, аутентификация, авторизация, аудит

Идентификация (*identification*) заключается в сообщении пользователем системы своего идентификатора, в то время как **аутентификация** (*authentication*) — это процедура доказательства пользователем того, что он есть тот, за кого себя выдает, в частности, доказательство того, что именно ему принадлежит введенный им идентификатор.

В процедуре аутентификации участвуют две стороны: одна сторона доказывает свою аутентичность, предъявляя некоторые доказательства, а другая сторона — **аутентификатор** — проверяет эти доказательства и принимает решение. В качестве доказательства аутентичности используются самые разные приемы:

- аутентифицируемый может продемонстрировать знание некоего общего для обеих сторон секрета: слова (пароля) или факта (даты и места события, прозвища человека и т. п.);
- аутентифицируемый может продемонстрировать, что он владеет неким уникальным предметом (физическими ключом), в качестве которого может выступать, например, электронная магнитная карта;
- аутентифицируемый может доказать свою идентичность, используя собственные биохарактеристики: рисунок радужной оболочки глаза или

отпечатки пальцев, которые предварительно были занесены в базу данных аутентификатора.

Сетевые службы аутентификации строятся на основе всех этих приемов, но чаще всего для доказательства идентичности пользователя применяются **пароли**.

Простота и логическая ясность механизмов аутентификации на основе паролей в какой-то степени компенсирует известные слабости паролей. Это, во-первых, возможность раскрытия и разгадывания паролей, а во-вторых, возможность «подслушивания» пароля путем анализа сетевого трафика. С целью снижения уровня угрозы раскрытия паролей администраторы сети, как правило, применяют встроенные программные средства для формирования политики назначения и использования паролей, включая задание максимального и минимального сроков действия пароля, хранение списка уже использованных паролей, управление поведением системы после нескольких неудачных попыток логического входа и т. п. Перехват паролей по сети можно предупредить путем их шифрования перед передачей в сеть. Тем не менее пароли являются наиболее уязвимым звеном сетевой безопасности, так как, зная пароль, всегда можно выдать себя за другого (рис. 5.11).



Рис. 5.11. «В Интернете никто не узнает,
что ты собака» (рис. Питера Штайнера)

Легальность пользователя может устанавливаться по отношению к различным системам. Так, работая в сети, пользователь может проходить процедуру аутентификации и как локальный пользователь, который претендует на ресурсы только данного компьютера, и как пользователь сети, который хочет

получить доступ ко всем сетевым ресурсам. При локальной аутентификации пользователь вводит свои идентификатор и пароль, которые автономно обрабатываются операционной системой, установленной на данном компьютере. При логическом входе в сеть данные о пользователе (идентификатор и пароль) передаются на сервер, который хранит учетные записи обо всех пользователях сети. Многие приложения имеют собственные средства определения легальности пользователя. И тогда пользователю приходится проходить дополнительные этапы проверки.

В качестве объектов, требующих аутентификации, могут выступать не только пользователи, но и различные устройства, приложения, текстовая и другая информация. Например, пользователь, обращающийся с запросом к корпоративному серверу, должен доказать ему свою легальность, но он также должен убедиться сам, что ведет диалог действительно с сервером своего предприятия. Другими словами, сервер и клиент должны пройти процедуру взаимной аутентификации. Здесь мы имеем дело с аутентификацией на уровне приложений. При установлении сеанса связи между двумя устройствами также часто предусматриваются процедуры взаимной аутентификации на более низком, канальном уровне. Аутентификация данных означает доказательство целостности этих данных, а также то, что они поступили именно от того человека, который объявил об этом. Для этого используется механизм электронной подписи.

В вычислительных сетях процедуры аутентификации часто реализуются теми же программными средствами, что и процедуры авторизации. В отличие от аутентификации, призванной выявлять легальных и нелегальных пользователей, система авторизации имеет дело только с **легальными** пользователями, которые уже успешно прошли процедуру аутентификации.

Средства **авторизации** (*authorization*) контролируют доступ легальных пользователей к ресурсам системы, предоставляя каждому из них именно те права, которые определены ему администратором. Помимо предоставления пользователям прав доступа к каталогам, файлам и принтерам, система авторизации может контролировать возможность выполнения пользователями различных системных функций, таких как локальный доступ к серверу, установка системного времени, создание резервных копий данных, выключение сервера и т. п.

Процедуры авторизации реализуются программными средствами, которые могут быть встроены в операционную систему или в приложение, а также поставляться в виде отдельных программных продуктов.

Аудит (*auditing*) — это фиксация в системном журнале событий, связанных с доступом к защищаемым системным ресурсам. Подсистема аудита современных ОС позволяет дифференцированно задавать перечень интересующих администратора событий с помощью удобного графического

интерфейса. Средства учета и наблюдения обеспечивают возможность обнаружить и зафиксировать важные события, связанные с безопасностью, или любые попытки создать, получить доступ или удалить системные ресурсы. Аудит призван фиксировать даже неудачные попытки «взлома» системы.

Учет и наблюдение означает способность системы безопасности «шпионить» за выбранными объектами и их пользователями и выдавать сообщения тревоги, когда кто-нибудь пытается читать или модифицировать системный файл. Если кто-то выполняет действия, определенные системой безопасности для отслеживания, то система аудита пишет сообщение в журнал регистрации, идентифицируя пользователя. Системный менеджер может создавать отчеты о безопасности, содержащие информацию из журнала регистрации. Для «сверхбезопасных» систем предусматриваются аудио- и видеосигналы тревоги на машинах администраторов, отвечающих за безопасность.

Поскольку никакая система безопасности не гарантирует защиту на уровне 100 %, последним рубежом в борьбе с нарушениями оказывается система аудита. 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. Эта информация, возможно, позволит найти злоумышленника или, по крайней мере, предотвратить повторение подобных атак путем устранения уязвимых мест в системе защиты.

Технология защищенного канала

Как уже было сказано, задачу защиты данных можно разделить на две подзадачи: защиту данных внутри компьютера и защиту данных в процессе их передачи из одного компьютера в другой. Для обеспечения безопасности данных при их передаче по публичным сетям используются различные технологии защищенного канала.

Технология защищенного канала призвана обеспечивать безопасность передачи данных по открытой транспортной сети, например по Интернету. 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.

Совокупность защищенных каналов, созданных предприятием в публичной сети для объединения своих филиалов, часто называют **виртуальной частной сетью¹** (Virtual Private Network, VPN).

Существуют разные реализации технологии защищенного канала, которые, в частности, могут работать на разных уровнях модели OSI. Так, функции популярного протокола **SSL** соответствуют уровню *представления* модели OSI. Протокол **IPSec** предусматривает все функции — взаимную аутентификацию, шифрование и обеспечение целостности, которые по определению свойственны защищенному каналу, а фирменный протокол туннелирования **PPTP** компании Microsoft защищает данные на *канальном* уровне.

Политика безопасности

Организация служб безопасности сети требует тщательной проработки **политики информационной безопасности**, которая включает несколько базовых принципов.

- ❑ *Предоставление каждому сотруднику предприятия того минимального уровня привилегий* на доступ к данным, который необходим ему для выполнения его должностных обязанностей.
- ❑ *Использование комплексного подхода к обеспечению безопасности.* Система защиты с многократным резервированием средств безопасности повышает вероятность сохранности данных. Так, например, физические средства защиты (закрытые помещения, блокировочные ключи) ограничивают непосредственный контакт пользователя только приписанным ему компьютером, встроенные средства сетевой ОС (система аутентификации и авторизации) предотвращают вход в сеть нелегальных пользователей, а для легального пользователя ограничивают возможности только разрешенными для него операциями (подсистема аудита фиксирует его действия).
- ❑ *Наличие единого контрольно-пропускного пункта.* Весь входящий во внутреннюю сеть и выходящий во внешнюю сеть трафик должен проходить через единственный узел сети, например, через **межсетевой экран**, или **брандмауэр** (firewall). Только это позволяет в достаточной степени контролировать трафик. В противном случае, когда в сети имеется множество пользовательских станций, имеющих независимый выход во внешнюю сеть, очень трудно скоординировать правила, ограничивающие права пользователей внутренней сети по доступу к серверам внешней сети и обратно — права внешних клиентов по доступу к ресурсам внутренней сети.
- ❑ *Баланс надежности защиты всех уровней* (при наличии многоуровневой системы защиты). Если в сети все сообщения шифруются, но ключи легкодоступны, то эффект от шифрования нулевой. Если внешний трафик

¹ О виртуальных частных сетях смотрите также раздел «Технология MPLS» главы 4.

сети, подключенной к Интернету, проходит через мощный брандмауэр, но пользователи имеют возможность связываться с узлами Интернета по коммутируемым линиям, используя локально установленные модемы, то деньги (как правило, немалые), потраченные на брандмауэр, можно считать выброшенными на ветер.

- *Использование таких средств безопасности, которые при отказе переходят в состояние максимальной защиты.* Это касается самых различных средств. Если в сети имеется устройство, которое анализирует весь входной трафик и отбрасывает кадры с определенным заранее заданным обратным адресом, то при отказе оно должно полностью блокировать вход в сеть. Неприемлемым следовало бы признать устройство, которое при отказе пропускало бы в сеть весь внешний трафик.
- *Баланс возможного ущерба от реализации угрозы и затрат на ее предотвращение.* Ни одна система безопасности не гарантирует защиту данных на уровне 100 %, поскольку является результатом компромисса между возможными рисками и возможными затратами. Определяя политику безопасности, администратор должен взвесить величину ущерба, которую может понести предприятие в результате нарушения защиты данных, и соотнести ее с величиной затрат, требуемых на защиту этих данных. Так, в некоторых случаях можно отказаться от дорогостоящего межсетевого экрана в пользу стандартных средств фильтрации обычного маршрутизатора, в других же можно пойти на беспрецедентные затраты. Главное, чтобы принятое решение было обосновано экономически.

Рекомендуемая литература

При составлении данного списка мы ограничились лишь теми книгами, в которых последовательно изложены основные концепции, определяющие современное состояние и тенденции развития компьютерных сетей и Интернета, то есть книгами, являющимися, по сути, учебниками. Мы добавили к этому списку книги такого же плана по операционным системам, так как реализации сетевых протоколов представляют собой программные модули операционных систем конечных узлов сети — пользовательских компьютеров и серверов, и промежуточных узлов сети — маршрутизаторов и коммутаторов. Конечно, существует большое количество хороших книг, посвященных отдельным технологиям или определенным узким аспектам этих технологий, но мы не стали включать их в наш список литературы, так как такое детальное описание не соответствует стилю и задачам данного учебника.

В качестве общей рекомендации тем, кто желал бы получить более полную информацию «из первых рук», мы советуем обращаться к стандартам, в частности к RFC. Эти документы находятся в свободном доступе на сайте www.rfc-editor.org. Для поиска в архиве описания того или иного протокола или концепций, принятых в сетях TCP/IP, необходимо либо знать номер соответствующего документа RFC, либо пользоваться ключевыми словами. Как отправную точку для поиска подробной информации можно задействовать Википедию — энциклопедию в Интернете (www.ru.wikipedia.org или более полный вариант на английском языке www.wikipedia.org). Помещенные там ссылки на стандарты дают читателю быстрый и непосредственный доступ к первоисточникам.

1. Камер, Дуглас (*Douglas E. Comer*). Сети TCP/IP. Том 1. Принципы, протоколы и структура. Издательство: Вильямс, 2003. 880 с. ISBN 5-8459-0419-6, 0-13-018380-9.
2. Камер, Дуглас (*Douglas E. Comer*). Принципы функционирования Интернета. Учебный курс (The Internet Book). СПб.: Питер, 2002. 384 с. ISBN 5-318-00464-4, 0-13030-8-528.
3. Камер, Дуглас (*Douglas E. Comer*). Сети TCP/IP. Том 1. Принципы, протоколы и структура. М.: Вильямс, 2003. 880 с. ISBN 5-8459-0419-6, 0-13-018380-9.

4. Камер, Дуглас (*Douglas E. Comer*). Принципы функционирования Интернета. Учебный курс (The Internet Book). СПб.: Питер, 2002. 384 с. ISBN 5-318-00464-4, 0-13030-8-528.
5. Олифер В., Олифер Н. Компьютерные сети. Принципы, технологии, протоколы. Учебник для вузов. СПб.: Питер, 2007. 960 с. ISBN 5-469-00504-6, 5-469-00504-9.
6. Олифер В. Г., Олифер Н. А. Сетевые операционные системы. Учебник для вузов. СПб.: Питер, 2008. 672 с. ISBN 978-5-91180-528-9.
7. Stevens, W. Richard. The Protocols (TCP/IP Illustrated, Volume 1). Addison-Wesley Professional, 1993. 600 с. ISBN 0201633469.
8. Столлингс, Вильям (*William Stallings*). Компьютерные сети, протоколы и технологии Интернета. Computer Networking with Internet Protocols and Technology. СПб.: БХВ-Петербург, 2005. 832 с. ISBN 5-94157-508-4.
9. Столлингс, Вильям (*William Stallings*). Операционные системы (Operating Systems. Internals and Design Principles). М.: Вильямс, 2004. 848 с. ISBN 5-8459-0310-6, 0-1303-1999-6.
10. Столлингс, В. (*William Stallings*). Передача данных (Business Data Communications). СПб.: Питер, 2004. 752 с. ISBN 5-94723-647-8, 0-13088-263-1.
11. Столлингс, Вильям (*William Stallings*). Компьютерные системы передачи данных (Data and Computer Communications). М.: Вильямс, 2002. 928 с. ISBN 5-8459-0311-4, 0-1308-4370-9.
12. Таненбаум, Э. (*A. Tanenbaum*). Архитектура компьютера (Structured Computer Organization). СПб.: Питер, 2007. 848 с. ISBN 5-469-01274-3, 0-13-148521-0.
13. Таненбаум, Э. (*A. Tanenbaum*). Компьютерные сети (Computer Networks). СПб.: Питер, 2007. 992 с. ISBN 978-5-318-00492-6, 5-318-00492-6.
14. Tanenbaum, Andrew S. Modern Operating Systems. Prentice Hall, 2007. 1104 с. ISBN 0136006639.
15. Халсалл, Фред. Передача данных, сети компьютеров и взаимосвязь открытых систем. Data Communications, Computerr Networks and Osi. М.: Радио и связь, 1995. 408 с. ISBN 5-256-0006002, 0-201-18244-0.
16. Halsall, Fred. Computer Networking and the Internet (5th Edition). Addison Wesley, 2005. 832 с. ISBN 0321263588.

**ПРЕДСТАВИТЕЛЬСТВА ИЗДАТЕЛЬСКОГО ДОМА «ПИТЕР»
предлагают эксклюзивный ассортимент компьютерной, медицинской,
психологической, экономической и популярной литературы**

РОССИЯ

Москва м. «Электрозводская», Семеновская наб., д. 2/1, корп. 1, 6-й этаж;
тел./факс: (495) 234-3815, 974-3450; e-mail: sales@piter.msk.ru

Санкт-Петербург м. «Выборгская», Б. Сампсониевский пр., д. 29а;
тел./факс (812) 703-73-73, 703-73-72; e-mail: sales@piter.com

Воронеж Ленинский пр., д. 169; тел./факс (4732) 39-43-62, 39-61-70;
e-mail: pitervrn@comch.ru

Екатеринбург ул. Бебеля, д. 11а; тел./факс (343) 378-98-41, 378-98-42;
e-mail: office@ekat.piter.com

Нижний Новгород ул. Совхозная, д. 13; тел. (8312) 41-27-31;
e-mail: office@nнов.piter.com

Новосибирск ул. Станционная, д. 36;
тел./факс (383) 350-92-85; e-mail: office@nsk.piter.com

Ростов-на-Дону ул. Ульяновская, д. 26; тел. (8632) 69-91-22, 69-91-30;
e-mail: piter-ug@rostov.piter.com

Самара ул. Молодогвардейская, д. 33, литер А2, офис 225; тел. (846) 277-89-79;
e-mail: pitvolga@samtel.ru

УКРАИНА

Харьков ул. Суздальские ряды, д. 12, офис 10-11; тел./факс (1038067) 545-55-64,
(1038057) 751-10-02; e-mail: piter@kharkov.piter.com

Киев пр. Московский, д. 6, кор. 1, офис 33; тел./факс (1038044) 490-35-68, 490-35-69;
e-mail: office@kiev.piter.com

БЕЛАРУСЬ

Минск ул. Притыцкого, д. 34, офис 2; тел./факс (1037517) 201-48-79, 201-48-81;
e-mail: office@minsk.piter.com

 Ищем зарубежных партнеров или посредников, имеющих выход на зарубежный рынок.
Телефон для связи: **(812) 703-73-73.**
E-mail: fuganov@piter.com

 Издательский дом «Питер» приглашает к сотрудничеству авторов.
Обращайтесь по телефонам: **Санкт-Петербург – (812) 703-73-72,**
Москва – (495) 974-34-50.

 Заказ книг для вузов и библиотек: **(812) 703-73-73.**
Специальное предложение – e-mail: kozin@piter.com



Основы компьютерных сетей



Профессиональные биографии Виктора и Натальи Олифер очень похожи. Оба они получили свое первое высшее образование в МВТУ им. Н. Э. Баумана (специальность «Электронные вычислительные машины»), а второе — в МГУ им. М. В. Ломоносова (специальность «Прикладная математика»).

После защиты диссертации каждый из них совмещал преподавание в вузах с научно-исследовательской работой. В 1995 году Наталья и Виктор стали читать лекции по сетевым технологиям в Центре информационных технологий при МГУ. Ими были разработаны несколько авторских курсов, которые и составили в дальнейшем основу для написания популярных учебников «Сетевые операционные системы» и «Компьютерные сети», а также книги «Computer Networks: Principles, Technologies and Protocols for Network Design».

В настоящее время Наталья Олифер работает независимым консультантом в области сетевых технологий, а Виктор Олифер участвует в проекте по развитию сети JANET, объединяющей университеты и исследовательские центры Великобритании.

В этой книге компактно, без углубления в детали, рассматриваются основные технологии компьютерных сетей. Читатель найдет здесь описание технологий как проводных, так и беспроводных локальных сетей, а также «горячие» технологии *MPLS* и *Carrier Ethernet*, созданные специально для построения глобальных сетей. Особое место в книге занимает раздел «Сети TCP/IP», который, по сути, представляет собой краткое введение в технологию Интернета. В издании также рассматриваются популярные сетевые службы и сервисы, такие, например, как электронная почта, веб-служба *WWW*, средства обеспечения безопасности сети.

Среди потенциальных читателей этой книги авторы прежде всего видят студентов, для которых курс «Компьютерные сети» является либо вводным, либо непрофицирующим. Книга может быть полезна тем специалистам, чья профессиональная деятельность связана с сетями лишь косвенно, например прикладным программистам, менеджерам по продажам сетевого оборудования и тем руководителям, которым нужны базовые знания о компьютерных сетях.



Заказ книг:

197198, Санкт-Петербург, а/я 619
тел.: (812) 703-73-74, postbook@piter.com

61093, Харьков-93, а/я 9130
тел.: (057) 758-41-45, 751-10-02, piter@kharkov.piter.c
www.piter.com — вся информация о книгах и веб-магазин

Основы компьютерных сетей
Учебное пособие (12)

Кировоград
К Марка, 51
(0522) 35-20-16



820013411
Ціна:
72.50 грн

12638

21.05.2009