Часть I

# Основы организации сетей

В этой части книги рассматриваются основы организации сетей. Она является одним из немногих разделов книги, которые не посвящены исключительно описанию оборудования Cisco. Но здесь приведена важная информация, поскольку для понимания сложных тем, которые рассматриваются в следующих частях, необходимо твердо усвоить основные принципы. В эту часть книги входят главы по сетевым моделям, а также по технологиям локальных и распределенных сетей. Такая информация окажется весьма полезной при изучении остальной части книги, в которой рассматривается применение описанных здесь принципов в устройствах Cisco. Кроме того, материал, изложенный в этих главах, позволит понять организацию любой сетевой среды, а не только той, которая состоит в основном из устройств Cisco. Короче говоря, здесь рассматривается технология, лежащая в основе организации сетей.

Глава 1

# Модель OSI

*В этой главе...*

Модель OSI (Open Systems Interconnection — взаимодействие открытых систем) не так проста, как кажется на первый взгляд. Она была первоначально предназначена для обеспечения разработки протоколов, не зависящих от конкретных поставщиков оборудования, и для получения возможности создания наборов протоколов вместо монолитных программ сетевой связи, но в настоящее время модель OSI фактически редко используется для таких целей. Но эта модель все еще имеет одно важное назначение: на данный момент она представляет собой одно из лучших инструментальных средств описания и классификации сложных последовательностей действий, которые происходят в сетях. Поскольку основная часть применяемых в наши дни наборов протоколов (например TCP/IP) была разработана с использованием другой модели, многие протоколы этих наборов не полностью соответствуют модели OSI, и это вызывает определенную путаницу. Например, в некоторых книгах утверждается, что протокол маршрутной информации (Routing Information Protocol — RIP) работает на сетевом уровне, а в других указано, что он работает на прикладном уровне. Однако в действительности этот протокол не принадлежит полностью только к одному из этих уровней. Он, как и многие другие, включает функции, относящиеся к обоим уровням. Из этого можно сделать вывод, что устранить такую путаницу можно только с помощью модели OSI, которая позволяет изучать сетевые операции и определять, на каком уровне они выполняются.

Основная цель изучения модели OSI в этой книге состоит в том, чтобы читатель мог понять, какие функции выполняются тем или иным устройством, просто узнав, к какому уровню относится данное устройство. Например, если известно, что физическая адресация, или управление доступом к передающей среде (Media Access Control — MAC), осуществляется на уровне 2, а логическая (IP-адресация) — на уровне 3, то читателю сразу же станет ясно, что коммутатор Ethernet, который отвечает за фильтрацию MAC-адресов (физических адресов), является прежде всего устройством уровня 2. Кроме того, встретив в книге утверждение, что маршрутизатор выполняет задачу определения маршрута на уровне 3, читатель уже будет иметь полное представление о том, какие действия выполняет маршрутизатор.

Именно поэтому в этой главе отведено определенное место описанию модели OSI. По тем же причинам читателю следует продолжить чтение этой главы, даже если он уверен, что достаточно хорошо знает модель OSI. Очень важно четко уяснить назначение этой модели, прежде чем переходить к освоению других тем.

## Общее определение термина “пакет”

Для описания фрагментов информации, передаваемых по сети, применяются термины *пакет*, *дейтаграмма*, *фрейм*, *сообщение* и *сегмент*. Все они по сути имеют один и тот же смысл, но относятся к разным уровням модели OSI. Например, пакет можно рассматривать как конверт с письмом. Чтобы отправить этот конверт по почте, необходимо выполнить ряд требований (рис. 1.1), которые перечислены ниже.

- **Подготовить почтовое вложение.** Эта составляющая почтового отправления представляет собой письмо, например, с фотографией новорожденного сына, отправляемой дяде Джо.
- **Написать на конверте адрес отправителя**. Эта составляющая служит в качестве обратного адреса, который должен быть написан на стандартном конверте. Адрес указывает, от кого поступило сообщение, и необходим даже просто на тот случай, если возникнут проблемы с доставкой письма.
- **Написать на конверте адрес получателя**. Эта составляющая представляет собой адрес дяди Джо, без которого письмо невозможно доставить намеченному получателю.
- **Пройти через систему проверки.** Эта составляющая представляет собой штемпель на почтовой марке. Он подтверждает, что письмо отправлено с соблюдением всех требований и соответствует стандартам почтовой службы.

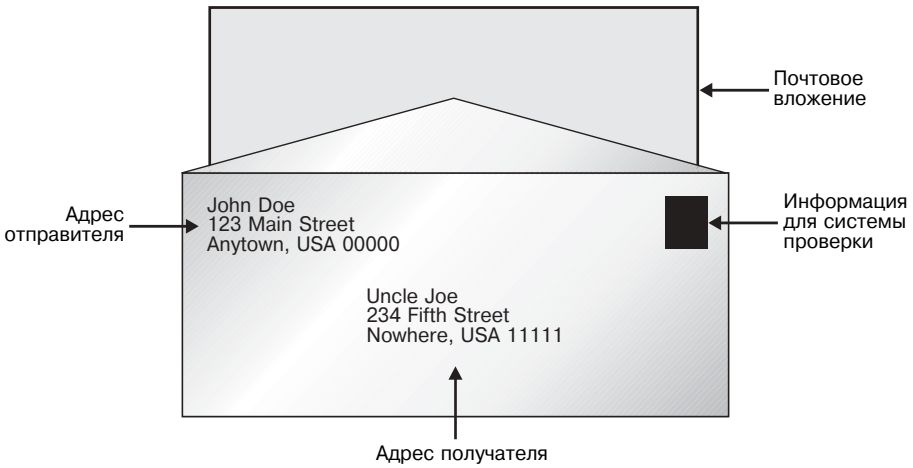


*Рис. 1.1. Обязательные составляющие обычного письма*

Передача сетевого пакета фактически происходит по таким же принципам, как и отправка обычного письма. Рассмотрим в качестве примера сообщение электронной почты, которое показано на рис. 1.2. Для его доставки адресату необходимо такая же информация, как и для обычного письма (а также некоторые другие компоненты, которые рассматриваются в данной главе); эта информация описана ниже.

- **Почтовое вложение.** Этот компонент представляет собой передаваемые данные, допустим, электронное письмо дяде Джо с сообщением о рождении сына.
- **Адрес отправителя.** Этот компонент служит в качестве обратного адреса для электронного письма. Он позволяет узнать, от кого поступило сообщение, даже просто на тот случай, если возникнет проблема при доставке электронной почты.
- **Адрес получателя.** Этот компонент представляет собой адрес электронной почты дяди Джо и необходим для правильной доставки электронной почты.
- **Информация для системы проверки.** Если речь идет о пакете, то этот компонент представляет собой определенную информацию для системы контроля ошибок. В данном случае применяется контрольная последовательность фрейма (Frame Check Sequence — FCS). Такую последовательность можно рассматривать как результат вычислений, выполненных над содержимым пакета с помощью некоторой математической формулы. Если вычисления FCS в пункте назначения (на компьютере дяди Джо) дадут правильный результат, это будет означать, что данные в пакете являются действительными и должны быть приняты. А если результаты вычислений окажутся неправильными, сообщение будет отброшено.

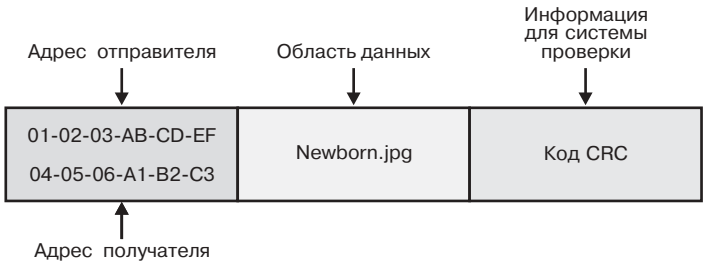


*Рис. 1.2. Основные компоненты пакета*

В следующих главах понятие пакета применяется для иллюстрации процесса прохождения данных сверху вниз по уровням модели OSI, затем по физическому кабелю, а после этого снизу вверх по уровням модели OSI, пока они не поступят в виде нового сообщения во входной почтовый ящик дяди Джо.

## Основы модели OSI

Модель OSI представляет собой один из способов многоуровневой организации сетей. В той или иной реализации набора протоколов некоторые из уровней модели могут даже не использоваться, но модель OSI разработана так, чтобы любую сетевую функцию можно было представить на одном из ее семи уровней. Описание уровней, начиная с уровня 7 и заканчивая уровнем 1, приведено в табл. 1.1. Здесь принята именно такая последовательность описания уровней, поскольку она позволяет лучше понять устройство модели.

***Таблица 1.1. Уровни модели OSI***

| Уровень | Назначение |
|---|---|
| Уровень 7 (прикладной — *application*) | Этот уровень отвечает непосредственно за взаимодействие с самим приложением. Он позволяет разработать приложение, используя для него минимальный объем сетевого кода. В приложении достаточно предусмотреть передачу прикладному протоколу информации о том, какие действия он должен выполнить, а прикладной протокол сам преобразует полученный запрос в команды, выполняемые набором протоколов |
| Уровень 6 (представительский — *presentation*) | На этом уровне выполняются все действия, которые связаны с форматированием пакета: сжатие, шифрование, кодирование и преобразование символов. Например, если текст в письме, полученном по электронной почте, представляет собой полную бессмыслицу, это означает, что возникла проблема на представительском уровне |
| Уровень 5 (сеансовый — *session*) | На этом уровне устанавливаются соединения (или сеансы) между двумя оконечными точками связи (обычно приложениями). Он обеспечивает настройку в приложении, находящемся на другом конце соединения, правильных параметров, позволяющих установить двухстороннюю связь с приложением-отправителем |
| Уровень 4 (транспортный — *transport*) | Этот уровень обеспечивает взаимодействие двух прикладных программ. В зависимости от применяемого протокола, на этом уровне могут выполняться функции обнаружения и устранения ошибок, установки и разрыва сеанса транспортного уровня, мультиплексирования, фрагментации и управления потоком данных |
| Уровень 3 (сетевой — *network*) | Этот уровень отвечает в основном за логическую адресацию и определение маршрута (или маршрутизацию) между группировками логических адресов |
| Уровень 2 (канальный — *datalink*) | Этот уровень отвечает за физическую адресацию и управление сетевой интерфейсной платой (которую называют также просто *сетевой платой*). В зависимости от применяемого протокола, на этом уровне может также осуществляться управление потоком данных. Кроме того, на этом уровне в пакет вводится последовательность FCS, в результате чего появляется возможность обнаруживать некоторые ошибки |
| Уровень 1 (физический — *physical*) | Этот уровень является самым простым, и выполняемые в нем функции в основном касаются физических характеристик сетевого соединения: кабельной разводки, соединителей и всех прочих физических компонентов. Этот уровень отвечает также за преобразование битов и байтов (логических единиц и нулей) в физическую форму (электрические импульсы, синусоидальные колебания или оптические сигналы) со стороны отправителя и за обратное преобразование в биты со стороны получателя |

При передаче данных по сети с одного компьютера на другой осуществляется такой процесс: данные исходят из приложения, передаются вниз по уровням модели, проходят через передающую среду (чаще всего это медный или волоконно-оптический кабель) в виде электрического или оптического сигнала, представляющего отдельные логические нули и единицы, после чего поднимаются по уровням модели на другом конце соединения. По мере выполнения этих действий на каждом уровне, который имеет соответствующий протокол, к пакету добавляется заголовок, указывающий способ обработки пакета на другом конце соединения с помощью такого же протокола. Этот процесс называется *инкапсуляцией данных*. Схема этого процесса приведена на рис. 1.3. На этой схеме AH обозначает заголовок прикладного уровня (*Application Header*), PH — представительского (*Presentation Header*), SH — сеансового (*Session Header*), TH — транспортного (*Transport Header*), NH — сетевого (*Network Header*), DH — канального (*Datalink Header*) и PH — физического (*Physical Header*). После прибытия к месту назначения пакет проходит вверх по уровням модели и на каждом уровне удаляются заголовки соответствующих протоколов. Ко времени поступления пакета в приложение в нем остаются только данные, которые принято также называть *содержимым пакета* (payload).

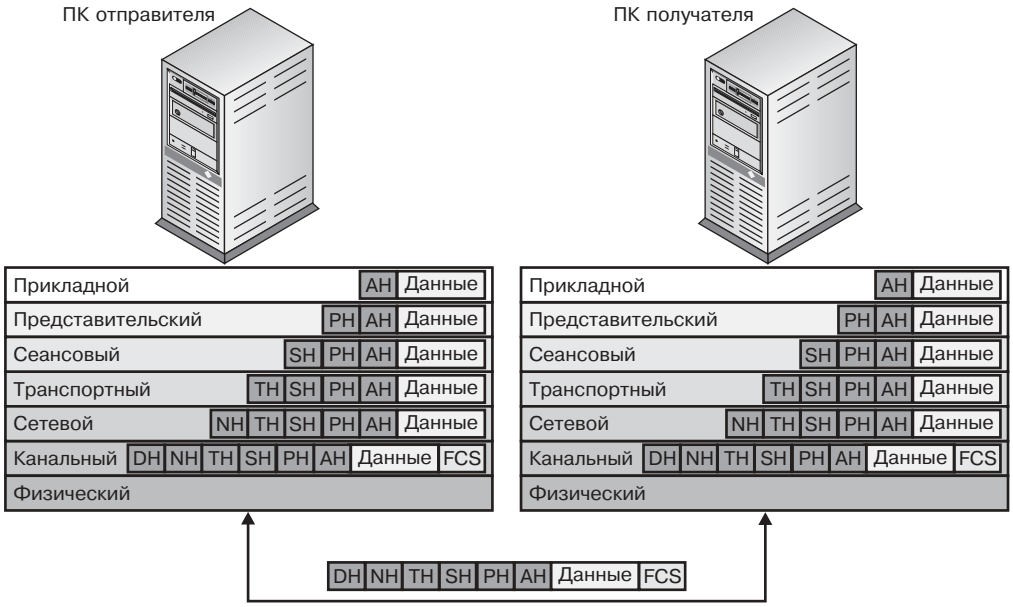


*Рис. 1.3. Инкапсуляция данных по мере прохождения пакета по уровням модели*

Перейдем к рассмотрению особенностей каждого уровня и дополнительных процессов, за которые отвечает каждый уровень.

## Уровень 7 — прикладной

Прикладной уровень отвечает за взаимодействие с пользовательским приложением. Но следует отметить, что обычно он обменивается данными не с самим пользовательским приложением, а, скорее, с сетевыми приложениями, которые применяются в пользовательском приложении. Например, при просмотре ресурсов Web пользовательским приложением является программа броузера, такая как Microsoft Internet Explorer. А в качестве сетевого приложения в данном случае используется программное обеспечение протокола HTTP, которое применяется также во многих других пользовательских приложениях (таких как Netscape Navigator). В общем, можно считать, что прикладной уровень отвечает за создание первоначального пакета, поэтому, если создается впечатле-

ние, что программное обеспечение протокола создает пакеты, которых до сих пор не существовало, то оно обычно относится к протоколу прикладного уровня. Хотя такое правило не всегда соблюдается (поскольку собственные пакеты создаются также протоколами, которые существуют на других уровнях), это общее определение протокола прикладного уровня вполне приемлемо. К числу широко применяемых протоколов прикладного уровня относятся HTTP, FTP, Telnet, TFTP, SMTP, POP3 и MAP. Дополнительные сведения о протоколах HTTP, FTP, SMTP и POP3 приведены в главе 5.

## Уровень 6 — представительский

Назначение представительского уровня понять проще всего, поскольку протокол этого уровня можно легко увидеть в действии. На представительском уровне происходит модификация формата данных. Например, к сообщению электронной почты может прилагаться изображение. Но простой протокол электронной почты (Simple Mail Transfer Protocol — SMTP) может обеспечить передачу только простого текста (состоящего из семибитовых символов в коде ASCII). Для обеспечения передачи изображения приложение должно воспользоваться протоколом представительского уровня для преобразования изображения в обычный текст. В данном случае применяется протокол многоцелевых почтовых расширений Internet (Multipurpose Internet Mail Extensions — MIME). Этот протокол отвечает также за обратное преобразование текста в изображение после его прибытия к месту назначения. Если эта работа не будет выполнена, то содержимое сообщения будет выглядеть примерно так, как это показано ниже.

```
BCNHS ^%CNE (37NC UHD^Y 3cNDI U&">{}I__DIwifd YYYTY TBVBC
```

Такая последовательность знаков, безусловно, не похожа на графическое изображение, и ее получение свидетельствует о наличии проблемы. Тем самым подтверждается сказанное выше, что обычно проще всего обнаружить наличие проблемы на представительском уровне. Кроме того, представительский уровень отвечает за сжатие и шифрование, а также за выполнение многих других действий (таких как эмуляция терминала), которые приводят к изменению формата данных. К числу наиболее широко применяемых форматов представления данных относятся ASCII, JPEG, MPEG и GIF.

## Уровень 5 — сеансовый

В отличие от предыдущих, работу протоколов сеансового уровня понять сложнее всего. Эти протоколы отвечают за установление, поддержание и завершение сеансов. Но это определение является слишком общим и расплывчатым, поскольку в установлении, поддержании и завершении сеансов в той или иной степени фактически участвуют и протоколы других уровней. Проще всего можно представить себе назначение сеансового уровня в том, что он выполняет функции посредника между двумя приложениями. Но как описано в главе 5, в наборе протоколов TCP/IP эта функция обычно выполняется на транспортном уровне, поэтому предыдущее утверждение не всегда соответствует истине. К числу наиболее широко применяемых протоколов сеансового уровня относятся RPC, LDAP и служба сеансов NetBIOS.

## Уровень 4 — транспортный

На транспортном уровне выполняется целый ряд функций. Наиболее важными из них являются контроль ошибок, их исправление и управление потоком данных. Транспортный уровень отвечает за надежную работу служб межсетевой передачи данных, функции которой выполняются незаметно для программ более высокого уровня. Проще всего можно понять, как осуществляются функции контроля и исправления ошибок на транспортном уровне, изучив различия между связью с установлением и без установления логического соединения.

## Связь с установлением и без установления логического соединения

Связь с установлением логического соединения получила такое название потому, что она предусматривает установление соединения между двумя компьютерами, подключенными к сети (называемыми также *хостами*), еще до начала передачи данных пользователем. Это позволяет обеспечить двухстороннюю связь. Иными словами, вначале протокол транспортного уровня предусматривает передачу получателю специальных пакетов, с помощью которых другой участник соединения может определить, что к нему вскоре поступят данные. Затем получатель передает специальный пакет отправителю, чтобы он мог узнать, что его "предупреждающее" сообщение получено. Такой предварительный обмен пакетами позволяет обоим участникам соединения убедиться в том, что связь между ними возможна.

В большинстве случаев связь с установлением логического соединения предусматривает также гарантии доставки. Иными словами, если при передаче пакета удаленному хосту происходит ошибка, то на транспортном уровне выполняется повторная передача этого же пакета, а если это невозможно, отправитель получает сообщение, что доставка пакета окончилась неудачей.

С другой стороны, связь без установления логического соединения обладает прямо противоположными свойствами. Во-первых, первоначально не устанавливается какое-либо соединение. Во-вторых, в большинстве случаев (но не во всех) не применяются какие-либо средства исправления ошибок. Обязанности по исправлению ошибок должно взять на себя само приложение или программное обеспечение протокола одного из уровней, находящихся выше или ниже транспортного уровня. Специалисты по сетям часто называют связь без установления логического соединения связью по принципу "отправить и забыть". По сути, протокол транспортного уровня отправляет пакет и "забывает" о нем.

В большинстве случаев уловить различие между протоколами с установлением и без установления логического соединения очень легко. Эти различия аналогичны тому, как отличаются друг от друга способы доставки обычного и заказного писем. Послав обычное письмо, отправитель может лишь надеяться, что оно поступит к адресату. У него нет возможности сразу же узнать, получено ли отправленное им сообщение. Это — связь без установления логического соединения. С другой стороны, при отправке заказного письма сообщение либо доставляется правильно и отправитель получает уведомление о вручении, либо предпринимаются неоднократные попытки его доставить, пока это сообщение не устаревает, и почтовая служба отказывается от дальнейших попыток; но отправитель получает уведомление и в этом случае. Так или иначе, отправитель уверен в том, что он узн*а*ет обо всем, что произошло, и сможет принять соответствующие меры. Это — типичная связь с установлением логического соединения.

## Управление потоком данных

В своей простейшей форме управление потоком данных представляет собой метод обеспечения того, чтобы чрезмерно интенсивный поток данных не захлестнул оконечную станцию. Например, предположим, что персональный компьютер A обрабатывает данные со скоростью 100 Мбит/с, а компьютер B — со скоростью 10 Мбит/с. Если компьютер A начнет передавать компьютеру B какие-то данные на полной скорости, то 90% этой информации будет потеряно, поскольку компьютер B не способен принимать информацию на скорости 100 Мбит/с. В предотвращении этой ситуации и состоит назначение средств управления потоком данных.

Применяемые в настоящее время методы управления потоком данных подразделяются на три типа, как описано в следующих разделах.

### Буферизация

По-видимому, самым простым из этих методов является буферизация, которая в основном применяется в сочетании с другими методами управления потоком данных. Буфер можно рассматривать как резервуар. Предположим, что из одной трубы

в этот резервуар втекает четыре литра воды в минуту, а из другой трубы, подключенной к резервуару, вода вытекает, но со скоростью только три литра воды в минуту. Если крышка резервуара открыта, что произойдет с лишней водой, если трубы, через которые поступает и вытекает вода, будут опущены в неглубокий поддон? Правильно — поддон быстро заполнится и вода польется на пол. То же самое происходит с данными, поступающими с компьютера A, который рассматривается в предыдущем примере. Для выхода из подобной ситуации можно применить такое же решение, как и в гидравлике — для приема лишней воды поставить "резервуар", или буфер. Но очевидно, что такое решение приводит к появлению других проблем. Прежде всего, буфер не может иметь бесконечный объем. Он позволяет легко справляться с временным увеличением объема трафика, но если поток данных, превышающий возможности приемного устройства, движется непрерывно, резервное пространство в конечном итоге целиком заполнится и в этот момент снова возникнет та же проблема — биты данных начнут бесследно исчезать.

### Уведомление о заторе

Метод с использованием уведомления о заторе является немного более сложным по сравнению с буферизацией и обычно используется в сочетании с буферизацией для устранения ее основных недостатков. При использовании метода с уведомлением о заторе после того, как буфера приемного устройства начинают заполняться (или явные проявления затора в сети обнаруживаются с помощью некоторых иных методов), приемная станция отправляет передающей станции сообщение, которое по сути означает "замедлить передачу данных". После того как буфер немного разгрузится, приемная станция может отправить другое сообщение с указанием, что передача может быть возобновлена. Очевидным недостатком такого решения является то, что при наличии в цепочке промежуточных устройств (таких как маршрутизаторы) уведомления о заторе лишь усугубляют ситуацию, заполняя буфера на каждом маршрутизаторе вдоль этой цепочки.

Например, предположим, что маршрутизатор A передает пакеты маршрутизатору C через маршрутизатор B (как показано на рис. 1.4). Как только буфер маршрутизатора C начинает заполняться, он передает уведомление о заторе маршрутизатору B. Это сообщение приводит к заполнению буфера маршрутизатора B. Затем маршрутизатор B отправляет уведомление о заторе маршрутизатору A. Это приводит к заполнению буфера маршрутизатора A, что в конечном итоге вызывает потерю данных (безусловно, этого не произойдет, если передающая станция определит, в чем смысл уведомлений о заторе, и полностью прекратит передачу данных). В конечном итоге маршрутизатор C перешлет маршрутизатору B сообщение о том, что может быть возобновлена передача, но к этому времени часть пакетов уже будет потеряна.

### Применение окон

Метод с применением окон представляет собой наиболее сложную и гибкую форму управления потоком данных и в настоящее время, вероятно, является одним из наиболее широко применяемых методов управления потоком данных. При передаче с применением окон разрешается передавать одновременно заранее согласованное количество пакетов (называемое *окном*) до получения подтверждения от приемной станции. Это означает, что возможность передачи одной станцией такого объема данных, который не может быть принят другой станцией, почти полностью исключена. Дело в том, что передающая станция, отправив разрешенное количество пакетов, должна дождаться ответа от удаленной приемной станции и только после этого отправить дополнительные данные. Метод передачи с применением окон используется не только для управления потоком данных, но и для устранения ошибок, как описано в главе 5.

К числу наиболее широко применяемых протоколов транспортного уровня относятся TCP, UDP и SPX, которые описаны более подробно в главах 5 и 7.

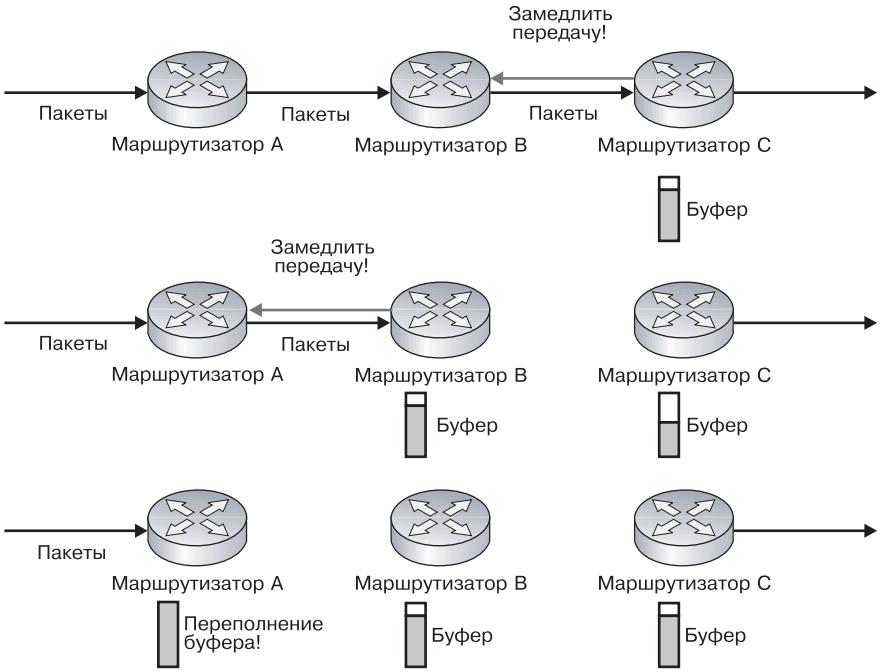


*Рис. 1.4. Проблемы, связанные с буферизацией и отправкой уведомлений о заторах*

## Уровень 3 — сетевой

Протоколы сетевого уровня обеспечивают логическую адресацию и определение маршрута (маршрутизацию). Методы логической адресации зависят от набора протоколов, но основные принципы остаются одинаковыми. Адреса сетевого уровня применяются в основном для указания местонахождения хоста. Эта задача обычно решается путем разделения адреса на две части: поле группы и поле хоста. Вместе эти поля полностью описывают хост, но лишь в контексте группы, к которой он относится. Такое разделение адреса позволяет каждому хосту учитывать только наличие других хостов в его группе и применять для передачи пакетов от одной группы к другой специализированные устройства, называемые *маршрутизаторами.*

К числу широко применяемых протоколов сетевого уровня относятся IP и IPX, которые описаны в главах 5—7.

## Уровень 2 — канальный

Канальный уровень предусматривает выполнение таких функций, как устранение коллизий, физическая адресация, распознавание ошибок и фреймирование, как описано в следующих разделах.

### Устранение коллизий

Методы устранения коллизий позволяют определить, как должен быть организован доступ к одному каналу передачи данных, если к нему подключено несколько хостов, которые пытаются одновременно использовать его для передачи. При полудуплексной широкополосной передаче без устранения коллизий нельзя обойтись, поскольку в применяемой при этом сетевой среде в любой момент времени только одно устрой-

ство может успешно передавать электрический сигнал. А если в этой среде попытки передачи будут предприняты одновременно двумя устройствами, то сигналы от этих устройств смешаются и возникнет так называемая *коллизия*. Такое явление, вероятно, лучше всего проиллюстрировать на рисунке (рис. 1.5).

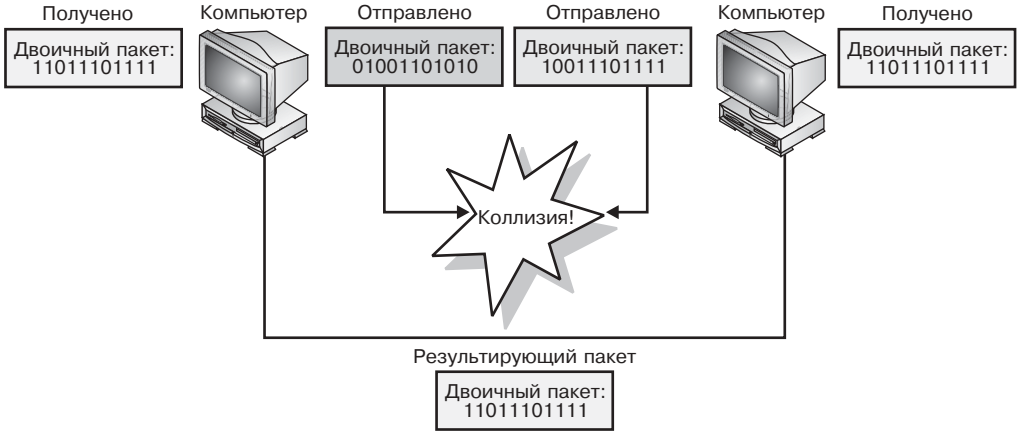


*Рис. 1.5. Коллизия и появление ошибочного пакета*

## Физическая адресация

Все устройства должны иметь физический адрес. В технологиях локальной сети таковым обычно является *MAC-адрес*. Физический адрес формируется таким образом, чтобы он мог однозначно обозначить определенное устройство, позволяя отличить его от всех прочих устройств в мире. MAC-адрес (называемый также *адресом Ethernet*, *адресом локальной сети*, *физическим адресом*, *аппаратным адресом*, а также известный под многими другими названиями) представляет собой 48-битовый адрес, который обычно записывается в виде 12 шестнадцатеричных цифр, таких как `01-02-03-AB-CD-EF`. Первые шесть шестнадцатеричных цифр определяют изготовителя устройства, а последние шесть — отдельное устройство, выпущенное этим изготовителем. Структура MAC-адреса показана на рис. 1.6. По традиции принято говорить, что эти постоянные адреса "прошиваются" в сетевой плате. Тем не менее, хотя и достаточно редко, иногда обнаруживаются дубликаты MAC-адресов. Поэтому в настоящее время очень многие сетевые устройства имеют MAC-адреса с перестраиваемой конфигурацией. Но так или иначе, физический адрес определенного типа является обязательным компонентом пакета.

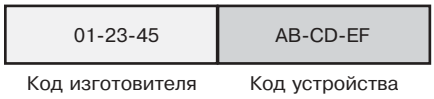


*Рис. 1.6. Структура MAC-адреса*

## Обнаружение ошибок

Еще одна функция канального уровня, обнаружение ошибок, позволяет определить, не произошло ли искажение пакета во время передачи. Для этого перед отправкой пакета на удаленный компьютер к нему добавляется *концевик* (так называется поле с контрольной суммой в конце пакета) с последовательностью FCS. Метод контроля с применением FCS предусматривает использование циклического избыточного кода (Cyclic Redundancy Check — CRC) для выработки цифрового значения и размещение этого значения в концевике пакета. После прибытия пакета к получателю извлекается значение поля FCS и снова применяется тот же алгоритм, с помощью которого было вычислено это первоначальное значение. Если пакет подвергся каким-либо изменениям, прежнее и новое значения FCS не совпадают, и пакет отбрасывается как ошибочный.

## Примечание

*Контроль с помощью FCS обеспечивает только обнаружение ошибок, но не их устранение. За устранение ошибок отвечает протокол более высокого уровня, как правило, транспортного.*

### Фреймирование

Термин *фреймирование* используется для описания организации элементов в пакете (пакет, передаваемый по сети, оформляется в виде фрейма). Эта задача является очень важной. Чтобы понять, с чем это связано, необходимо рассмотреть, как происходит передача данных физическим устройством. Прежде всего следует учесть, что все данные, передаваемые по кабелям сети, являются просто комбинацией битов `0` и `1`. Поэтому при получении устройством цепочки битов, такой как `01101010001010111101011110101010100101000101010111` и т.д., оно должно определить, какая часть этой цепочки соответствует MAC-адресу, данным или последовательности FCS. Для этого требуется ключ. Физический формат пакета показан на рис. 1.7.


*Рис. 1.7. Пример показывает, как к потоку битов применяется ключ фреймирования Ethernet 802.3 для выделения отдельных частей этого потока*

Кроме того, поскольку существуют разные типы фреймов, в протоколах канального уровня на обоих взаимодействующих компьютерах должны использоваться фреймы одинаковых типов, так как лишь при этом условии получатель сможет определить, что фактически содержит полученный им пакет. Пример искажения, возникающего при нарушении формата фрейма, показан на рис. 1.8.

На этом рисунке значения длины полей принятого и фактически ожидаемого фрейма не совпадают. Данный пример показывает, что если один компьютер отправляет пакет в формате 802.3, а другой ожидает поступления пакета в формате протокола доступа к подсети (Sub-Network Access Protocol — SNAP), между ними невозможно установить взаимодействие, поскольку компьютеры безуспешно пытаются найти компоненты пакета, которые фактически представлены в другом формате.

К числу наиболее распространенных протоколов канального уровня относятся практически все протоколы 802 (802.2, 802.3, 802.5 и т.д.), LAPB, LAPD и LLC.

## Уровень 1 — физический

На физическом уровне выполняются наиболее важные функции передачи данных по сравнению со всеми другими уровнями. К физическому уровню относятся все соединители, кабели, спецификации частот, требования к расстояниям и задержкам при распространении сигналов, регламентируемые напряжения, короче говоря, все физические параметры.

К числу наиболее распространенных протоколов физического уровня относятся EIA/TIA 568A и 568B, RS 232, 10BaseT, 10Base2, 10Base5, 100BaseT и USB.

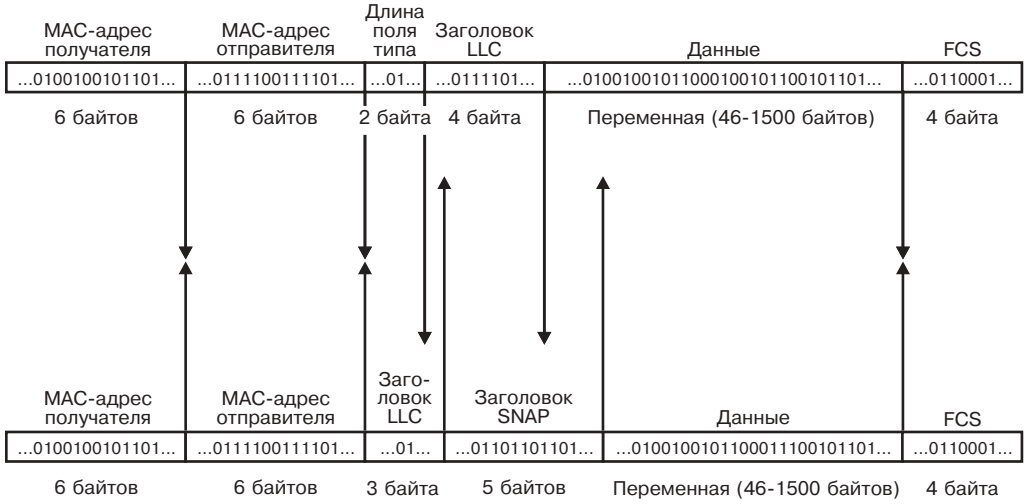


*Рис. 1.8. Несовпадение форматов полей из-за неправильного выбора типа фрейма*

### Одноранговая связь

Специалисты по сетям называют *одноранговой связью* процесс взаимодействия протокола каждого уровня на компьютере отправителя с соответствующим уровнем на компьютере получателя. Следует отметить, что одинаковые уровни не взаимодействуют непосредственно друг с другом, но обмен данными по сети организован так, как если бы они действительно напрямую связывались друг с другом. К пакету, передаваемому с одного хоста на другой, по мере прохождения по уровням сетевой модели добавляются все необходимые заголовки, а когда этот пакет после его приема снова проходит по уровням модели, но в противоположном направлении, информация в каждом заголовке пакета обрабатывается только тем уровнем, которому соответствует конкретный заголовок. Все остальное на этом уровне рассматривается как данные. Процесс снятия заголовков показан на рис. 1.9 (для обозначения заголовков применяются такие же сокращения, как и на рис. 1.3).

Необходимо еще раз подчеркнуть, что протоколом каждого уровня обрабатывается только заголовок, который относится точно к такому же уровню протокола на другом компьютере. Остальная часть пакета рассматривается им как данные (хотя фактически не является таковой). Поэтому можно считать, что каждый уровень протокола на одном компьютере взаимодействует с соответствующем ему уровнем на другом компьютере.

## Общее описание взаимодействия по сети

Наконец, рассмотрим, как происходит обмен данными по сети между двумя компьютерами на каждом уровне (рис. 1.9). Следует отметить, что этот пример формально не совсем точен. Он приведен только для иллюстрации, поскольку в нем показано, как выполняется определенная функция на каждом уровне, даже если ее выполнение в действительности происходит немного иначе. Основная техническая неточность в этой схеме допущена на сетевом уровне, где показано поле “Промежуточный адрес получателя”. В действительности поля промежуточного адреса не существует, но поскольку в этой книге еще не рассматривалась тема, как работают средства маршрутизации, на данный момент этот пример может служить вполне удачной иллюстрацией.


*Рис. 1.9. Заголовки, добавляемые на каждом уровне модели*

В данном примере передается электронная почта по протоколам TCP/IP. Передача сообщения начинается с уровня 7. К нему добавляется заголовок MAPI (Mail Application Programming Interface — интерфейс прикладного программирования для электронной почты). Затем пакет передается на представительский уровень, где происходит добавление заголовка MIME, с помощью которого получатель сможет определить формат сообщения. На сеансовом уровне происходит преобразование имен, и доменное имя `techtrain.com` преобразуется в IP-адрес `209.130.62.55`. На транспортном уровне все это сообщение, которое имеет длину 256 Кбайт, разбивается на четыре фрагмента по 64 Кбайт и устанавливается сеанс TCP с использованием метода окон для управления потоком данных. На сетевом уровне выполняется маршрутизация и пакет передается на ближайший маршрутизатор (который здесь обозначен с помощью поля промежуточного адреса назначения).

Следует также отметить, что на сетевом уровне (логические) IP-адреса преобразуются в (физические) MAC-адреса, чтобы с ними мог работать протокол более низкого уровня. На канальном уровне пакет снова фрагментируется, но на этот раз преобразуется во фреймы, которые соответствуют максимальной единице передачи данных (Maximum Transmission Unit — MTU) передающей среды. На физическом уровне данные передаются в виде электрических сигналов. Принятые данные снова проходят по уровням модели, но в обратном направлении. При этом выполняются действия, обратные тем, которые были выполнены на компьютере отправителя, и в конечном итоге пакет преобразуется в один фрагмент данных размером 256 КБайт в формате, приемлемом для соответствующего приложения.

## Другие сетевые модели

Важное значение с точки зрения организации сетей имеет также модель DoD (Department of Defense — Министерство обороны США), так как в основе протоколов TCP/IP лежит не модель OSI, а именно эта модель. Поскольку модель DoD во многом совпадает с моделью OSI, тот факт, что она является фундаментом протоколов TCP/IP, может привести к некоторой путанице при изучении модели OSI. Верхние уровни модели DoD не совпадают с верхними уровнями модели OSI, поэтому в разных книгах можно встретить различные описания порядка расположения протоколов в модели OSI. Но здесь необходимо прежде всего учитывать, что фактически *знание* того, где должен быть указанный протокол модели OSI, необходимо в основном для успешной сдачи экзаменов, а на практике важнее всего *понимание* назначения каждого уровня модели. Соответствие уровней моделей OSI и DoD показано на рис. 1.10.


*Рис. 1.10. Модели DoD и OSI*

Модели OSI и DoD позволяют наглядно представить процесс сетевого взаимодействия, а компания Cisco применяет в своей работе *иерархическую межсетевую модель*, которая представляет собой многоуровневое отображение топологического проекта объединенной сети. Эта модель разработана в целях максимального повышения производительности; в то же время она обеспечивает оптимальную отказоустойчивость. Применение этой модели позволяет упростить конструкцию сети путем распределения функций по уровням сетевого проекта. Очевидным недостатком данной модели в сетях небольших и средних размеров является высокая стоимость проекта, но если задача состоит в создании высокопроизводительной, масштабируемой, резервируемой объединенной сети, то применение такого подхода является одним из наилучших способов реализации в проекте поставленных целей.

Иерархическая межсетевая модель Cisco состоит из трех уровней.

- **Уровень ядра сети**. Этот уровень в объединенной сети соответствует опорной сети. Поскольку опорная сеть играет такую важную роль, любые серьезные нарушения в ее работе скорее всего будут заметны для всех, кто использует эту объ-

единенную сеть. Кроме того, поскольку скорость здесь играет очень важную роль (в связи с огромным объемом трафика, который проходит по опорной сети), на этом уровне практически не должны быть реализованы функции, требующие значительных ресурсов маршрутизации или коммутации. Иными словами, маршрутизация, обработка списков доступа, сжатие, шифрование и все прочие функции, требующие больших затрат ресурсов, должны быть выполнены до того, как пакет поступит в ядро сети.

- **Распределительный уровень**. Этот уровень занимает промежуточное положение между уровнем ядра сети и уровнем доступа. Клиенты не взаимодействуют непосредственно с этим уровнем, но на нем выполняется основная часть функций обработки передаваемых ими пакетов. На этом уровне выполняется также основная часть вспомогательных функций. В частности, на нем функционируют службы маршрутизации, обеспечения качества обслуживания (Quality of Service — QoS), проверки списков доступа, шифрования, сжатия и трансляции сетевых адресов (Network Address Translation — NAT).
- **Уровень доступа**. На этом уровне пользователям предоставляется доступ к локальным сегментам. Характерной особенностью уровня доступа является применение соединений локальной сети, обычно в сетевой среде небольшого масштаба (такой как отдельное здание). Иными словами, именно на этом уровне происходит подключение клиентов к сети. Обычно на уровне доступа выполняется коммутация Ethernet и другие основные функции.

Пример практического применения этой модели приведен на рис. 1.11.

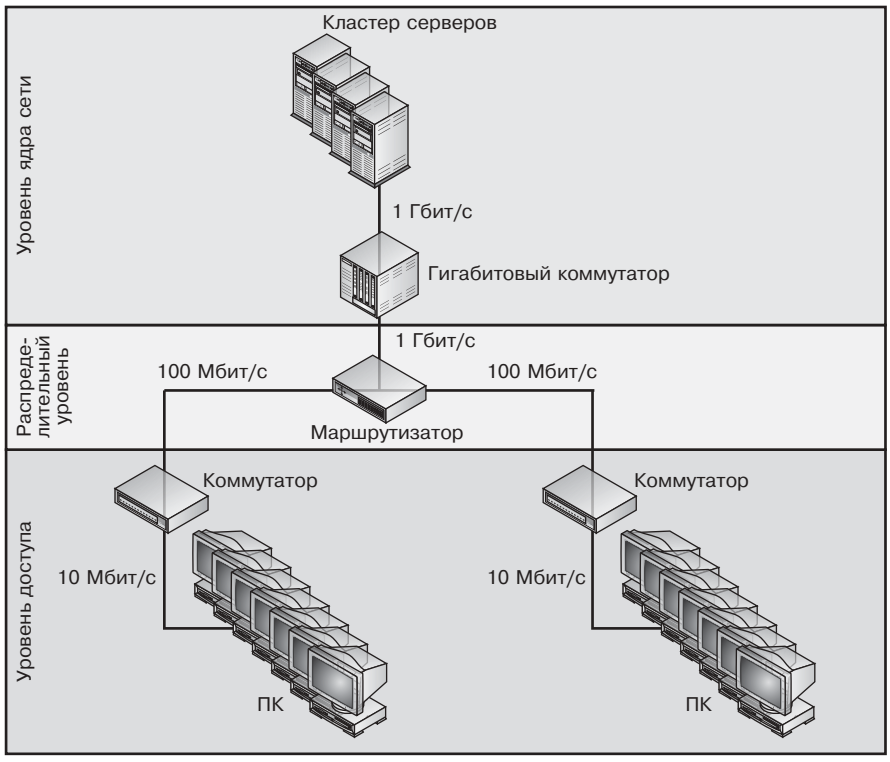


*Рис. 1.11. Иерархическая межсетевая модель Cisco*

## Резюме

В этой главе рассматривались наиболее широко применяемые сетевые модели, включая модели OSI, DoD и Cisco. Приведенная здесь информация позволяет лучше понять, о чем идет речь при описании различных тем на основе многоуровневого сетевого подхода, принятого в этой книге, а также должна служить в качестве руководства для понимания назначения маршрутизации и коммутации в любой среде.