Чарльз Р. Северанс

Как работают компьютерные сети и интернет

Introduction to Networking

How The Internet Works

Charles Severance

Illustrations: Mauro Toselli

Как работают компьютерные сети и интернет

Чарльз Р. Северанс

Иллюстрации: Мауро Тозелли

ДМК ПРЕСС
издательство
Москва, 2022

УДК 004.738, 004.62
ББК 32.973
С28

С28 Чарльз Р. Северанс
Как работают компьютерные сети и интернет / пер. с англ. П. М. Бомбаковой – М.: ДМК Пресс, 2022. – 116 с.: ил.

ISBN 978-5-97060-959-0

Цель данной книги – дать общее представление о технической структуре и архитектуре сети интернет. Подробно описываются разные уровни сетевой модели TCP/IP (канальный, сетевой, транспортный и прикладной); особое внимание уделяется безопасности транспортного уровня. Рассмотрена система доменных имен. В заключительной главе представлена семиуровневая модель взаимодействия открытых систем (OSI) в сравнении с моделью TCP/IP.

В конце каждой главы приводятся глоссарий по теме и контрольные вопросы, позволяющие читателям проверить свои знания.

Книга предназначена для широкой аудитории. Для чтения необязательны знания в области математики и технологий.

ISBN 978-5-97060-959-0 (рус.)

Оглавление

Предисловие от издательства

Отзывы и пожелания

Мы всегда рады отзывам наших читателей. Расскажите нам, что вы думаете об этой книге, – что понравилось или, может быть, не понравилось. Отзывы важны для нас, чтобы выпускать книги, которые будут для вас максимально полезны.

Вы можете написать отзыв прямо на нашем сайте **www.dmkpress.com**, зайдя на страницу книги, и оставить комментарий в разделе «Отзывы и рецензии». Также можно послать письмо главному редактору по адресу **dmkpress@gmail.com**, при этом напишите название книги в теме письма.

Если есть тема, в которой вы квалифицированы, и вы заинтересованы в написании новой книги, заполните форму на нашем сайте по адресу **http://dmkpress.com/authors/publish_book/** или напишите в издательство по адресу **dmkpress@gmail.com**.

Список опечаток

Хотя мы приняли все возможные меры для того, чтобы удостовериться в качестве наших текстов, ошибки все равно случаются. Если вы найдете ошибку в одной из наших книг – возможно, ошибку в тексте или в коде, – мы будем очень благодарны, если вы сообщите нам о ней. Сделав это, вы избавите других читателей от расстройств и поможете нам улучшить последующие версии этой книги.

Если вы найдете какие-либо ошибки в коде, пожалуйста, сообщите о них главному редактору по адресу **dmkpress@gmail.com**, и мы исправим это в следующих тиражах.

Нарушение авторских прав

Пиратство в интернете по-прежнему остается насущной проблемой. Издательство «ДМК Пресс» очень серьезно относятся к вопросам защиты авторских прав и лицензирования. Если вы столкнетесь в интернете с незаконно выполненной копией любой нашей книги, пожалуйста, сообщите нам адрес копии или веб-сайта, чтобы мы могли применить санкции.

Пожалуйста, свяжитесь с нами по адресу электронной почты **dmkpress@gmail.com** со ссылкой на подозрительные материалы.

Мы высоко ценим любую помощь по защите наших авторов, помогающую предоставлять вам качественные материалы.

Предисловие

Цель данной книги – дать общее представление о технической структуре и архитектуре сети интернет. Книга предназначена для широкой аудитории. Так, она подойдет даже тем, кто не обладает техническим опытом или математическими навыками. Интернет в своем строении удивителен, и его должны понимать все, кто им пользуется.

Несмотря на то что эта книга не затрагивает напрямую сертификации Network+ или CCNA, я все же надеюсь, что знакомство с ней послужит неплохой стартовой точкой для тех, кто в будущем желает получить эти сертификаты.

Я хочу выразить благодарность Памеле Фокс (Pamela Fox) из Академии Хана за идею создания вводного курса сетевых технологий с использованием открытых материалов.

Изначально материал, положенный в основу этой книги, представлял собой недельную лекцию, которую я читал в Школе информации Мичиганского университета в рамках курса «Сетевые вычисления» (Networked Computing – SI502) начиная с 2008 года. Затем я внес в эту лекцию некоторые уточнения и расширил ее, превратив в трехнедельный курс «История, технология и безопасность интернета» (Internet History, Technology, and Security – IHTS), который, начиная с 2012 года, посетило более 100 000 студентов в рамках проекта Coursera. Для того чтобы превратить курс лекций в полноценную книгу, я добавил некоторые детали, благодаря чему теперь она может быть полезна при прохождении вводного курса, посвященного архитектуре интернета, а также ее можно читать просто ради удовольствия.

Особенность этой книги заключается в том, что она представляет собой продукт сотрудничества с моими друзьями Мауро Тозелли (Mauro Toselli) (**@xlontrax**) и Сью Блюменберг (Sue Blumenberg). С Мауро и Сью я познакомился в 2012 году. Тогда они были волонтерами в Community Teaching Assistants (CTAs) для моего курса IHTS (Internet History, Technology, and Security) в Coursera. За последние три года мы стали коллегами и хорошими друзьями. К слову, это отличный пример того, как открытое образование объединяет людей.

Дополнительные материалы к этой книге можно найти по адресу **http://www.net-intro.com/**.

Если вам понравилась книга или же, напротив, вы нашли в ней ошибки, свяжитесь со мной в Twitter.

Чарльз Р. Северанс (**@drchuck**)
www.dr-chuck.com
Анн-Арбор, Мичиган, США
20 мая 2015 года

Глава 1

Введение

На первый взгляд использование сети интернет выглядит крайне просто. Мы переходим на какой-либо веб-адрес, и открывается нужная страница. Или же мы заходим в нашу любимую социальную сеть и просматриваем фотографии друзей, семейные фото или снимки с домашними животными. Однако за кажущейся простотой скрывается большое количество сложного программного и аппаратного обеспечения. Разработка технологий, на основе которых работает современный интернет, началась в 1960-х годах. Так, созданию первого «интернета» в 1980-х годах в рамках проекта NSFNet предшествовали 20 лет исследования технологии межсетевого взаимодействия. С тех пор благодаря исследованиям и разработкам в области сетевых технологий сети стали больше, быстрее, и теперь могут связывать миллиарды устройств по всему миру.

С целью лучше понять, как работает интернет сегодня, рассмотрим с помощью каких технологий осуществлялась коммуникация людей и устройств на протяжении многих лет.

1.1. Общение на расстоянии

Представьте группу из пяти человек, сидящих в кругу. Для нас вполне естественно говорить с любым человеком в комнате, но только если все находящиеся в ней вежливы и не ведут разговоры параллельно друг с другом. Так, в этой ситуации людям просто нужно уметь слышать друг друга и координировать использование общего пространства в комнате.

Но что, если поместить этих людей в разные комнаты таким образом, чтобы они больше не могли видеть и слышать друг друга? Как пара людей сможет общаться друг с другом? Один из способов – проложить между ними провод с микрофоном на одном конце и динамиком на другом. Однако в таком случае остальные все еще будут слы-

шать их разговор, поэтому необходимость в ведении только одного разговора сохраняется.

Каждому человеку потребуется четыре динамика (для каждого потенциального собеседника) и большое количество проводов. Осуществить подобное для пяти человек – проблема, однако становится еще сложнее, если количество участников возрастает до сотни или тысячи.

Именно так выглядели первые телефонные системы 1900-х годов – провода, микрофоны, динамики. Поскольку о прокладке проводов между всеми телефонами не могло идти и речи, эти системы не позволяли всем людям объединиться в единую сеть. Каждый из них мог связаться только с *оператором*, который затем соединял два провода вместе, чтобы пара людей могла начать разговор. Когда разговор заканчивался, оператор отсоединял провода.

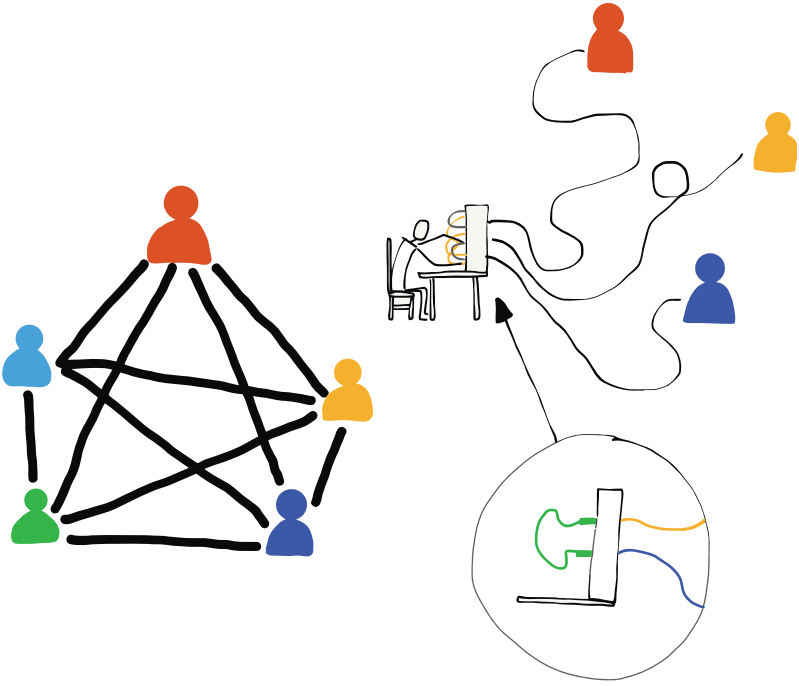

Рис. 1.1. Общение с помощью телефонных операторов

Первые локальные телефонные системы работали при условии, если дом или офис клиента был близко к зданию, где находится оператор, и между ними можно было проложить провод.

Но что, если в связи нуждались тысячи людей, находящихся на расстоянии сотен километров друг от друга? Разумеется, проложить 100-км линию связи от каждого дома к одному центральному офису оператора невозможно. Вместо этого телефонные компании создали систему с несколькими центральными офисами и проложили между ними телефонные линии. Так, при осуществлении коммуникации на больших расстояниях соединение может проходить через несколько центральных офисов. До появления оптоволоконных кабелей междугородные телефонные линии прокладывались через огромные столбы с большим количеством проводов. Количество проводов на столбах представляло количество возможных параллельных междугородных телефонных звонков.

Рис. 1.2. Столбы междугородной телефонной сети

По мере увеличения длины телефонных линий росла и их стоимость. Поэтому вышеописанные соединения, проходящие через несколько центральных офисов, были довольно дорогими и немногочисленными. На первых порах локальные звонки были относительно недорогими. Однако междугородные соединения были намного дороже и тарифицировались поминутно. Такая ценовая политика объяснялась тем, что во время разговора по междугородному телефону никто другой не мог воспользоваться этим же соединением. Телефонным компаниям были выгодны более короткие разговоры, благодаря

чему междугородные соединения быстрее становились доступными для других клиентов.

После появления волоконно-оптической связи телефонные компании начали использовать более продвинутые методы соединения клиентов, позволяющие передавать множество междугородных соединений по одной линии. Если на какой-либо старой фотографии вы видите множество проводов на одном столбе, скорее всего, это телефонные линии, которые использовались не для передачи электричества.

1.2. Компьютеры общаются иначе

Когда люди разговаривают по телефону, сначала они совершают звонок, общаются какое-то время, а затем кладут трубку. Согласно статистике, большая часть времени уходит отнюдь не на сам разговор. По крайней мере, так было раньше, когда смартфоны были далеко не у всех. Однако процесс коммуникации у различных устройств, в том числе и приложений на вашем смартфоне, значительно отличается от человеческого. Иногда они посылают короткие сообщения для того, чтобы проверить, доступно ли другое устройство. Иногда отправляют сообщения среднего размера, например одно изображение или длинное сообщение электронной почты. Также они могут отправлять и большие сообщения, например целый фильм или часть программного обеспечения для установки. Загрузка таких файлов может занимать минуты или даже часы. Так, при взаимодействии между устройствами сообщения могут быть короткими, среднего размера или же длинными.

Изначально пары устройств также соединялись с помощью проводов. Самый простой способ отправить данные с одного устройства на другое – выстроить исходящие сообщения в очередь и посылать сообщения одно за другим с такой скоростью, с какой устройства и соединения могут передавать данные. В таком случае каждое сообщение ожидает своей очереди, а затем, после того как будут отправлены предшествующие сообщения, оно передается через соединение.

Если устройства находились в одном здании, их легко можно было соединить проводами. Если они находились в одном городе, обычно приходилось арендовать каналы у телефонных компаний. Владельцы устройств зачастую просили соединять линии в их центральном офисе, чтобы одному устройству не приходилось каждый раз устанавливать связь с другим. Такие арендованные *выделенные каналы* были лучшим решением, поскольку они всегда находились в активном режиме. Использовались они 24 часа в сутки и поэтому стоили довольно дорого.

В случае, если устройства находились в разных городах, выделенные каналы расширялись посредством прокладки дополнительных проводов, соединяющих центральные офисы. Поскольку соединений между центральными офисами было мало, междугородные выделенные каналы были дорогими, и их стоимость росла по мере увеличения длины проложенных линий. Однако, если у владельца было достаточно денег, он мог арендовать прямой канал между устройствами для обмена данными. Но такая система работала только между устройствами одного производителя, поскольку у каждого из них был собственный способ использования телефонных каналов для соединения устройств и отправки данных.

1.3. Ранние глобальные сети передачи данных с промежуточным хранением

В 1970–80-х годах исследователи, работающие в разных университетах мира, как никогда нуждались в возможности отправлять друг другу сообщения и данные с помощью таких соединений между устройствами. Однако стоимость каждого соединения была крайне высока и росла с увеличением расстояния. Именно поэтому зачастую устройства были подключены только к ближайшим машинам. Но, если устройство, к которому вы были подключены, было подключено к другому устройству, а это устройство в свою очередь – к третьему и т. д., появлялась возможность отправить сообщение на большее расстояние (при условии, что каждое из задействованных устройств сможет хранить и передавать ваши данные).

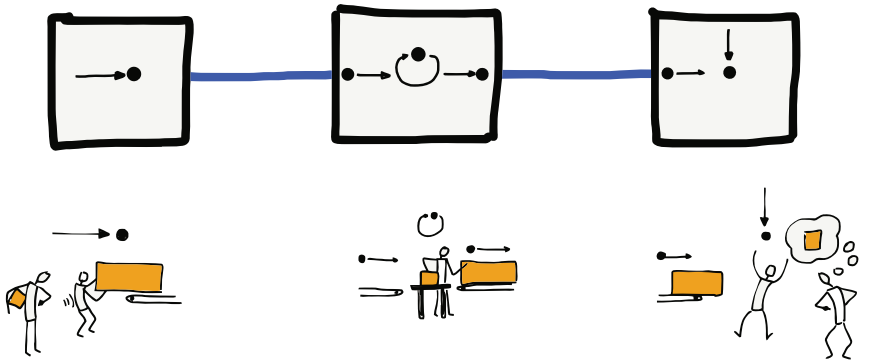

Рис. 1.3. Сети передачи данных с промежуточным хранением

Со временем при относительно небольшом количестве соединений появилась возможность отправлять данные на большие расстояния с помощью «лоскутного одеяла» сетевых соединений. Однако этот процесс занимал довольно много времени. Так, перед отправкой на следующее устройство по маршруту сообщение сталкивалось с большим количеством очередей. После доставки сообщения на промежуточное устройство оно хранилось некоторое время (которое варьировалось в зависимости от трафика и могло составлять до нескольких часов), а затем перенаправлялось на другое соединение (совершало *переход*).

Таким образом, отправка одного сообщения могла занимать минуты, часы или даже дни (в зависимости от трафика на каждом этапе маршрута). Однако это все же было намного быстрее, чем посылать в другую страну письмо или почтовую открытку.

1.4. Пакеты и маршрутизаторы

Самым важным нововведением, позволившим сообщениям быстрее перемещаться по сети с несколькими переходами, стало разбиение каждого сообщения на небольшие фрагменты, которые впоследствии отправлялись по отдельности. В области сетевых коммуникаций эти сообщения называются *пакетами*. Впервые эта идея была высказана в 1960-х годах, однако до 1980-х годов она не имела широкого распространения, поскольку ее реализация требовала большей вычислительной мощности и более сложного сетевого программного обеспечения.

При разбиении сообщения на отдельные пакеты, если короткое сообщение было отправлено после длинного сообщения, первому не приходилось ждать завершения отправки второго. Вместо этого первому пакету короткого сообщения требовалось дождаться лишь завершения отправки текущего пакета длинного сообщения. До тех пор, пока короткое сообщение не было отправлено полностью, а передача длинного сообщения не возобновилась по полному сетевому соединению, система чередовала отправку пакетов этих сообщений.

Разбиение сообщения на пакеты значительно сократило объем памяти, требующейся от промежуточных устройств. Теперь вместо того, чтобы хранить сообщения в течение нескольких часов, им достаточно было хранить некоторое количество пакетов лишь несколько секунд, а остальные пакеты ожидали своей очереди на исходящем канале.

По мере отказа от сетей с промежуточным хранением начали появляться особые устройства, специализирующиеся на транспортировке пакетов. Первоначально они назывались *интерфейсными процессорами обработки сообщений* (Interface Message Processors – IMPs)

и представляли собой интерфейсы между обычными устройствами и остальной частью сети. Позже эти устройства получили название *маршрутизаторов*, поскольку их цель заключалась в маршрутизации получаемых пакетов к конечному месту назначения.

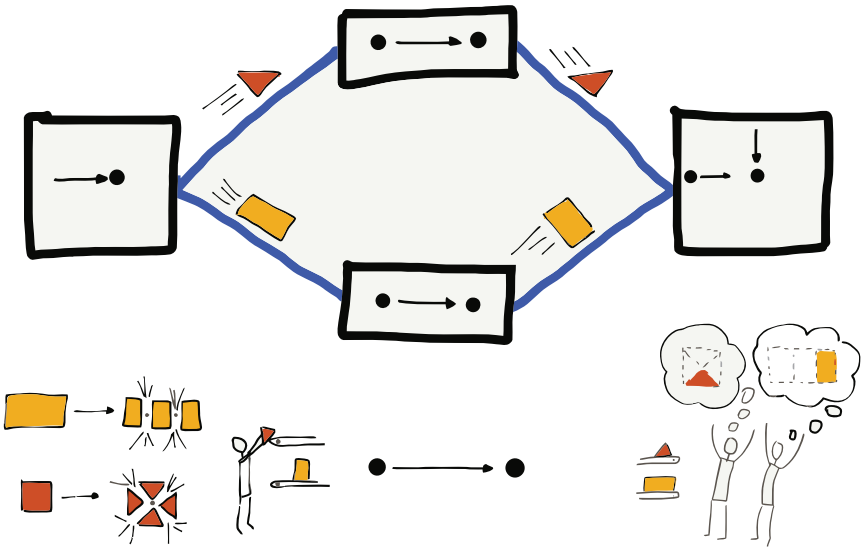

Рис. 1.4. Отправка пакетов

С появлением маршрутизаторов, специализирующихся на транспортировке пакетов по сетям с несколькими переходами, подключать устройства разных производителей к одной сети стало намного проще. Все, что для этого требовалось, – подключить обычное устройство к маршрутизатору. Остальную же работу по организации связи выполняли другие маршрутизаторы.

Если несколько устройств, находящихся радом, были объединены в *локальную сеть* (Local Area Network – LAN) с помощью физического провода, следовало подключить маршрутизатор к локальной сети. Затем, после отправки данных через маршрутизатор все устройства в этой сети могли отправлять данные дальше через *глобальную сеть* (Wide Area Network – WAN).

1.5. Пакеты и адресация

В первых сетях передачи данных с промежуточным хранением для каждого сообщения необходимо было иметь информацию об ис-

ходном и конечном устройствах. Каждый из них имел уникальный номер, который назывался *адресом*. Перед отправкой сообщения на другое устройство сначала необходимо было указать исходный и конечный адреса. В таком случае устройства, хранящие и передающие сообщения, могли выбрать для него оптимальный маршрут.

При разбиении длинного сообщения на отдельные пакеты для выбора оптимального маршрута исходный и конечный адреса должны были быть указаны для каждого из них. Помимо этого, для каждого пакета требовались данные, сообщающие «относительный адрес» или положение пакета в сообщении, что позволяло принимающему устройству собрать пакеты в правильном порядке и восстановить исходное сообщение.

1.6. Вывод

Объединив все вышеописанные аспекты, мы с легкостью сможем понять, как устроена современная сеть интернет. Итак, существуют специализированные устройства, называемые маршрутизаторами, которые отвечают за транспортировку пакетов от источника к месту назначения. На пути от исходного устройства к конечному каждый пакет проходит через несколько таких маршрутизаторов.

Несмотря на то что зачастую пакеты представляют собой части более крупного сообщения, маршрутизаторы пересылают каждый из них по отдельности, опираясь на указанные исходные и конечные адреса. Пакеты одного и того же сообщения могут проходить по разным маршрутам. Также иногда пакеты приходят не по порядку; более поздний пакет может прибыть раньше предыдущего в результате так называемых заторов. Каждый пакет содержит «относительный адрес», который в дальнейшем позволяет конечному устройству собрать пакеты в правильном порядке и восстановить исходное сообщение.

Благодаря созданию сетей с несколькими переходами общая стоимость связи внутри определенной географической области может распределяться между большим количеством связанных и отдельных пользователей. Обычно пакеты следуют по кратчайшему маршруту, однако, если соединение на этом маршруте перегружено или прервано, маршрутизаторы могут скооперироваться и перенаправить трафик. Таким образом, пакеты транспортируются по иным, в том числе более длинным маршрутам, в конечном итоге позволяющим доставить пакеты быстрее.

Итак, интернет строится на основе набора взаимодействующих маршрутизаторов, которые одновременно способны перемещать пакеты из разных источников в разные пункты назначения. Каждое устройство или локальная сеть подключены к маршрутизатору, который перенаправляет трафик в различные пункты назначения. Маршрутизатор может обрабатывать данные с одного устройства, например смартфона, с нескольких устройств в одном здании или с тысячи устройств, подключенных к сети университетского городка. Термин «интернет» происходит от термина *internetworking* (межсетевое взаимодействие), который подразумевает соединение множества сетей для совместной работы. Наши устройства подключаются к локальным сетям, а интернет соединяет локальные сети вместе, таким образом все подключенные устройства могут взаимодействовать друг с другом.

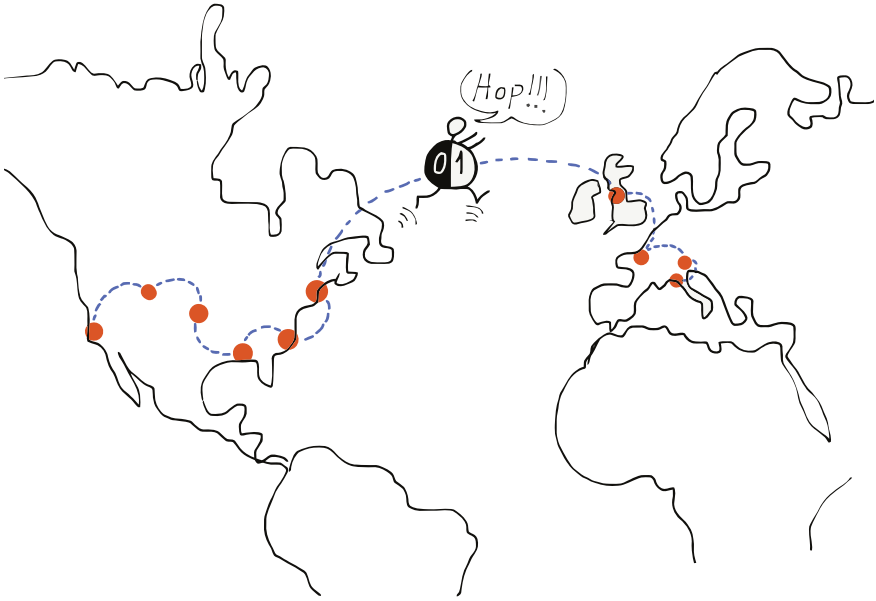

Рис. 1.5. Глобальное соединение

1.7. Глоссарий

Адрес – уникальный номер, который присваивается устройству и позволяет маршрутизировать сообщения на него.

Выделенный канал – постоянно активное соединение для отправки данных на большие расстояния, арендуемое у телефонной компании или другого подобного предприятия.

Глобальная сеть – сеть, покрывающая большие расстояния вплоть до возможности отправки сообщений по всему миру. Обычно такая сеть строится с использованием каналов связи, принадлежащих нескольким различным организациям.

Локальная сеть – сеть, охватывающая ограниченную возможностью прокладки проводов или мощностью радиопередатчика область.

Маршрутизатор – специализированное устройство, предназначенное для приема входящих пакетов по нескольким каналам и быстрой транспортировке пакетов по наиболее оптимальному исходящему каналу с целью ускорения их доставки.

Оператор (телефонный) – сотрудник телефонной компании, помогающий людям осуществлять телефонные звонки.

Переход – физический участок сети. Как правило, на пути от исходного к конечному устройству пакеты совершают несколько переходов.

Пакет – фрагмент более крупного сообщения ограниченного размера. Большие сообщения или файлы разбиваются на множество пакетов, каждый из которых впоследствии отправляется отдельно. Обычно максимальный размер пакета составляет от 1000 до 3000 знаков.

Сеть передачи данных с промежуточным хранением – сеть, в которой при отправке данных с одного устройства на другое сообщение сохраняется на промежуточном устройстве до тех пор, пока для него не станет доступным исходящее сетевое соединение (обычно этот период довольно долгий).

1.8. Контрольные вопросы

Вы можете пройти этот тест онлайн по адресу **http://www.net-intro.com/quiz/**.

1. Чем занимались первые телефонные операторы?
   А. Обслуживали вышки сотовой связи.
   Б. Соединяли пары проводов, чтобы люди могли разговаривать по телефону.
   В. Прокладывали медные провода между городами.
   Г. Сортировали пакеты по мере их поступления в место назначения.

2. Что такое выделенный канал?
   А. Граница между арендованным и находящимся в собственности телефонным оборудованием.
   Б. Кабель между клавиатурой и монитором.
   В. Провод, идущий от одного офиса телефонной компании к другому.
   Г. Постоянное телефонное соединение.

3. Как долго при сетевой передаче с промежуточным хранением сообщение может храниться на промежуточном компьютере?
   А. Менее секунды.
   Б. Не более четырех секунд.
   В. Менее минуты.
   Г. До нескольких часов.

4. Что такое пакет?
   А. Упаковка товаров для отправки.
   Б. Небольшая емкость для хранения.
   В. Часть более крупного сообщения, отправляемого по сети.
   Г. Объем данных, который можно было сохранить на перфокартах ранних версий.

5. Какое из приведенных ниже определений соответствует термину «маршрутизатор»?
   А. Сортировщик почты.
   Б. Холодильник.
   В. Скоростной поезд.
   Г. Кабель связи.

6. Как назывались первые сетевые маршрутизаторы?
   А. Процессоры межконфессиональных сообщений (Interfaith Message Processors).
   Б. Персептроны движения в интернете (Internet Motion Perceptrons).
   В. Программы мгновенного обмена сообщениями (Instant Message Programs).
   Г. Интерфейсные процессоры сообщений (Interface Message Processors).

7. Что еще требовалось для правильной маршрутизации каждого сегмента сообщения помимо разделения больших сообщений на более мелкие части?

А. Адрес источника и приемника в каждом сегменте сообщения.
Б. ID и пароль для каждого сегмента сообщения.
В. Маленькая батарея для хранения каждого сегмента сообщения.
Г. Небольшой блок отслеживания для поиска потерянных сообщений, например GPS.

8. Почему отправка сообщений по всему миру через интернет осуществляется практически бесплатно?

А. За все соединения платят правительства.
Б. Соединения оплачиваются за счет рекламы.
В. Ресурсами делятся и пользуются большое количество людей.
Г. Взимать плату за междугородные соединения незаконно.

Глава 2

Сетевая архитектура

При построении таких сложных систем, как интернет, инженеры стараются разбить подобные задачи на более мелкие, которые могут быть решены независимо друг от друга. Так, впоследствии они приходят к решению общей задачи. Именно поэтому инженеры, работавшие над первыми сетями, разделили основную задачу на четыре подзадачи (направления разработки), над которыми разные группы их коллег могли бы работать независимо.

Рис. 2.1. Четырехуровневая модель TCP/IP

Они дали этим четырем направлениям следующие названия: (1) канальный уровень (Link), (2) сетевой уровень (Internetwork), (3) транспортный уровень (Transport) и (4) прикладной уровень (Application). Обычно эти направления представляют в виде слоев, где канальный уровень является нижним, а прикладной – верхним. На канальном уровне осуществляется проводное и беспроводное соединение вашего устройства с локальной сетью. Прикладной же уровень – это то, с чем мы, пользователи, непосредственно взаимодействуем. Например, к архитектуре прикладного уровня относятся веб-браузеры.

Неофициально вышеописанную систему принято называть моделью TCP/IP. Такое название складывается из двух основных протоколов, на которых строится интернет, – *протокола управления передачей данных* (Transmission Control Protocol – TCP), относящегося к транспортному уровню, и интернет-протокола (IP), относящегося к сетевому уровню.

Мы кратко рассмотрим каждый из слоев, начиная с нижнего.

2.1. Канальный уровень

Канальный уровень отвечает за подключение вашего устройства к локальной сети и организацию передачи данных. На сегодняшний день канальный уровень успешно функционирует, в том числе и в беспроводной сети. Однако беспроводные устройства способны отправлять данные лишь на ограниченные расстояния. Так, смартфон обычно связывается с вышкой, находящейся в нескольких километрах от него. Если же вы, например, пользуетесь смартфоном в движущемся поезде, по мере следования устройству необходимо будет переключаться на новые вышки. Ноутбук, подключенный к сети Wi-Fi, как правило, обменивается данными с точкой доступа, находящейся в пределах 200 м.

Длина кабеля, обеспечивающего доступ к интернету для стационарного компьютера, обычно достигает не более 100 м. Также зачастую технологии канального уровня используются несколькими устройствами, которые могут находиться рядом друг с другом.

В локальных сетях совместного пользования на канальном уровне возникает два основных вопроса. Первый из них касается кодировки и отправки данных. Беспроводные соединения требуют использования одинаковых радиочастот и единой кодировки. В проводных соединениях необходимо наличие установленного уровня напряжения для передачи бита информации по проводной линии связи. Для канального уровня оптоволоконных соединений необходимо согласовать используемую частоту света и скорость отправки данных.

Второй вопрос касается взаимодействия с другими устройствами. Все они могут отправлять данные параллельно. Однако, если бы все устройства в сети отправляли данные сразу после их появления, возникали бы конфликты между сообщениями. В результате чего наступил бы хаос, и приемные станции смогли бы уловить только шум. Именно поэтому инженеры начали поиски способа, позволяющего соблюдать очередность работы станций в общей сети. Одним из

таких способов является разбиение больших сообщений на пакеты, которые затем отправляются по отдельности. Так, если в текущий момент данные отправляет только одно устройство, оно будет делать это настолько быстро, насколько это возможно. Однако в случае, если таких устройств три, каждое из них отправит один пакет, а затем будет ждать, пока два других отправят свои пакеты. После того как каждое другое устройство отправит по одному пакету, первое устройство отправит свой следующий пакет. Такая система позволяет распределять доступ к сети равномерно между всеми устройствами.

Но как одно устройство узнает о том, что другие тоже хотят отправить данные? Для решения этой проблемы была разработана гениальная технология под названием «множественный доступ с прослушиванием несущей и обнаружением конфликтов» (Carrier Sense Multiple Access with Collision Detection – CSMA/CD). Несмотря на длинное название, эта технология довольно проста. Когда ваше устройство хочет отправить данные, вначале оно проверяет, не отправляет ли другое устройство данные в сеть (проверка несущей частоты). Если ни один компьютер не отправляет данные, то ваше устройство начинает отправку. Во время отправки данных оно также контролирует состояние канала на предмет того, сможет ли он получить свои данные обратно. Если устройство обнаруживает, что это возможно, то оно делает вывод, что канал все еще свободен, и продолжает передачу. Но, если два устройства начали отправку примерно в одно и то же время, может возникнуть конфликт, вследствие которого ваше устройство не сможет получить собственные данные. При обнаружении конфликта оба устройства прекращают передачу, немного ждут и затем пробуют еще раз. Перед повтором передачи каждое устройство делает паузу разной продолжительности.

После отправки пакета ваше устройство уступает место для передачи другим отправителям. Если одно из них обнаруживает, что ваше устройство прекращает отправку, и начинает отправлять собственный пакет, устройство заметит использование канала другим устройством и дождется завершения этого процесса, после чего попытается отправить следующий пакет (множественный доступ).

Вышеописанная технология работает эффективно, когда данные хочет отправить только одно устройство или же когда несколько устройств хочет отправлять данные одновременно. В первом случае устройство может эффективно использовать общую сеть, отправляя пакеты один за другим, а во втором доступ к сети равномерно распределяется между устройствами.

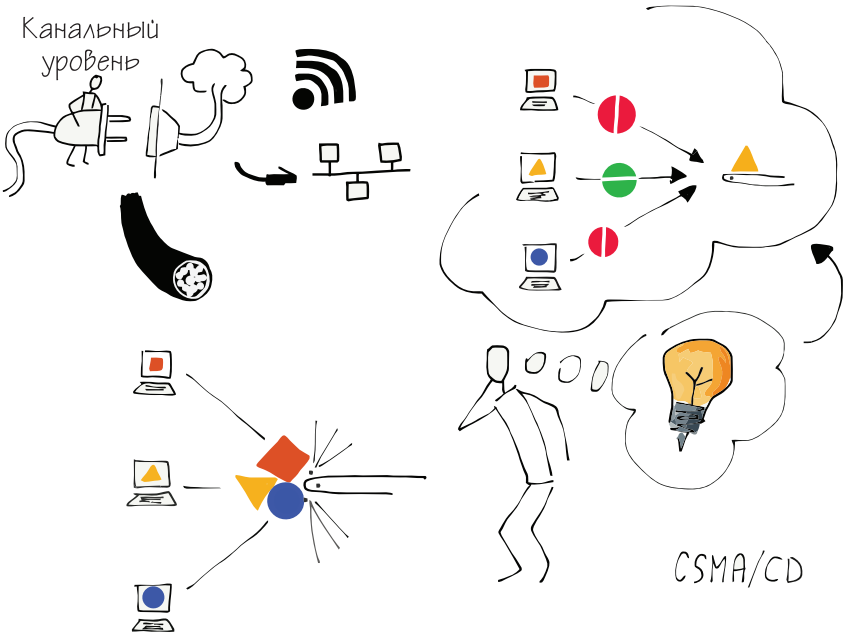

Рис. 2.2. Прослушивание несущей / обнаружение конфликтов

Некоторые канальные уровни, например сотовое соединение смартфона, Wi-Fi-соединение, спутниковый или кабельный модем, являются общими. Такие технологии, как CSMA/CD, помогают им обеспечивать равномерное распределение доступа к сети между устройствами. Другие же канальные уровни, например оптоволоконные кабели и выделенные линии, обычно не используются совместно и предназначены для установки соединений между маршрутизаторами. Однако такие соединения также являются элементами канального уровня.

Инженеры, работающие над технологиями канального уровня, особое внимание уделяют решению проблемы передачи данных несколькими устройствами по одному каналу, который может находиться на расстоянии от нескольких метров до сотен километров. Однако для транспортировки данных на столь протяженные расстояния устройствам необходимо отправлять пакеты через некоторое количество маршрутизаторов, соединенных несколькими канальными уровнями. Так, каждое перемещение пакета через другой канальный уровень от одного маршрутизатора к другому называется переходом. Для того чтобы отправить данные на другую сторону земного шара,

пакетам предстоит пройти через 20 маршрутизаторов или сделать 20 переходов.

2.2. Сетевой уровень (IP)

Как только пакет, предназначенный для отправки, проходит по первому соединению, он попадает в маршрутизатор. Затем маршрутизатор уточняет адрес назначения, содержащийся в пакете, и выстраивает наиболее оптимальный путь к месту назначения. Каждый маршрутизатор обрабатывает огромное количество пакетов, которые могут быть предназначены для миллиона разных устройств. Так, далеко не все маршрутизаторы могут определить точное местоположение и построить наилучший маршрут к каждому возможному конечному устройству. Поэтому обычно маршрутизатор старается найти лучший вариант маршрута доставки пакета ближе к месту назначения.

Каждый маршрутизатор, через который проходит пакет, также делает все возможное, чтобы осуществить доставку в место назначения. По мере прохождения пакетом своего пути маршрутизаторы начинают определять место его назначения все точнее. Как только пакет достигает последнего канала передачи данных на своем пути, канальный уровень точно определяет место назначения.

Данный процесс можно сравнить с планированием маршрута отпуска. В таких маршрутах обычно тоже много переходов. Представим, что первый переход – это поездка на машине, такси или автобусе до железнодорожного вокзала. Затем вы садитесь на пригородный поезд и следуете из вашего городка в большой город. Там вы садитесь на поезд дальнего следования и едете в большой город другой страны. Затем вы пересаживаетесь на пригородный поезд и отправляетесь в небольшую деревню, где и хотели отдохнуть. Выйдя из поезда, вы садитесь в автобус, а после автобуса идете к своей гостинице.

Если бы в поезде, который следует между большими городами, вы спросили бы у проводника, где находится отель в маленькой деревне, вряд ли он смог бы на это ответить. Задача проводника заключается в том, чтобы приблизить вас к месту назначения, и только это имеет значение, пока вы едете в поезде дальнего следования. При посадке в автобус в небольшой деревне можно спросить кондуктора, какая остановка находится ближе всего к вашей гостинице. И когда вы, наконец, выходите из автобуса на нужной остановке, то можете обратиться к прохожему с просьбой подсказать путь к гостинице.

Чем дальше вы находитесь от пункта назначения, тем меньше нужно знать точные сведения о том, как туда добраться. Когда вы далеко,

все, что вам нужно знать, – это как стать «ближе» к месту назначения. По этому же принципу работают маршрутизаторы. Точный путь знают только ближайшие к конечному устройству маршрутизаторы. Остальные нацелены на то, чтобы переместить ваше сообщение ближе к месту назначения.

Однако во время пути следования пакетов, как и во время путешествия, могут возникнуть проблемы или задержки, которые потребуют некоторых изменений в маршруте.

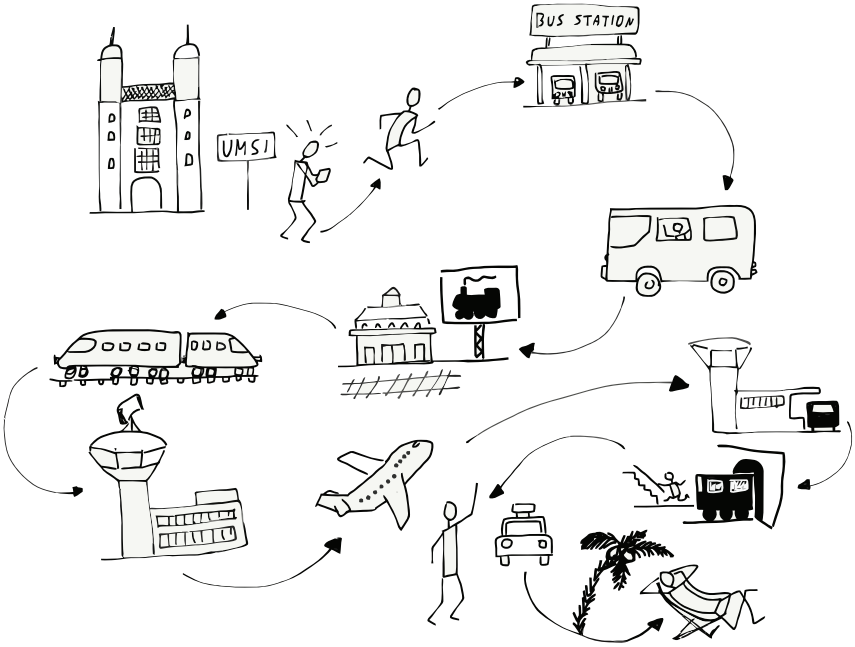

Рис. 2.3. Путешествие

Маршрутизаторы обмениваются специальными сообщениями, тем самым информируя друг друга о возникающих задержках трафика или сбоях в сети. Так, это обеспечивает своевременное перемещение пакетов со старого проблемного маршрута на новый. Также маршрутизаторы (которые, к слову, составляют ядро интернета) довольно умны и быстро адаптируются как к небольшим, так и к крупным перебоям или некорректной работе сетевых соединений. Иногда скорость соединения может снижаться из-за перегрузки. В других же случаях соединение может быть разорвано физически (когда, например, какая-нибудь строительная бригада по ошибке выкапывает кабель и

повреждает его). Иногда проблемы возникают в результате стихийных бедствий, таких как ураган или тайфун. В таких случаях маршрутизаторы и каналы отключаются повсеместно на определенной территории. Как правило, маршрутизаторы обнаруживают сбои достаточно быстро и по возможности сразу же меняют маршруты пакетов.

Однако бывают случаи, когда пакеты теряются. Такие проблемы решаются на следующем уровне интернет-архитектуры.

2.3. Транспортный уровень (TCP)

Сетевой уровень одновременно прост и сложен. Его задача заключается в определении маршрута пакета к конечному устройству.

Но бывает такое, что пакеты теряются, задерживаются или прибывают в пункт назначения не по порядку; так пакет, отправленный позже, нашел короткий путь быстрее, чем отправленный раньше. Каждый пакет содержит адрес исходного и конечного устройств, а также свой относительный адрес. Зная относительный адрес и размер каждого пакета, конечное устройство может восстановить исходное сообщение, даже если пакеты были получены не по порядку.

В процессе восстановления данных конечным устройством и доставки их в принимающее приложение периодически оно отправляет на исходное устройство уведомления о получении пакетов, в которых указывает, какая часть сообщения была получена и восстановлена. Если конечное устройство обнаруживает, что части восстановленного сообщения отсутствуют, в большинстве случаев это является следствием потери или задержки пакетов. В таком случае конечное устройство делает небольшую паузу, а затем отправляет исходному устройству запрос на повторную отправку данных, которые потенциально могли быть потеряны.

Исходное устройство хранит копии отправленных частей исходного сообщения до тех пор, пока конечное устройство не подтвердит их получение. Как только исходный компьютер получает подтверждение о получении части сообщения, оно отбрасывает соответствующие данные и затем отправляет новые.

Объем данных, отправляемых исходным устройством до начала ожидания подтверждения, называется *размером группы сетевых пакетов*. Если этот показатель имеет небольшое значение, передача данных замедляется, поскольку исходное устройство все еще ожидает подтверждения. Если исходное устройство отправляет слишком большое количество данных до начала ожидания подтверждения, ввиду перегрузки маршрутизаторов или линий междугородной свя-

зи могут возникнуть проблемы с трафиком. В таких случаях в начале выставляется небольшое значение размера группы сетевых пакетов, а также определяется время, необходимое для получения первых подтверждений. Если подтверждение осуществляется быстро, исходное устройство постепенно увеличивает размер группы сетевых пакетов. Если же подтверждение происходит медленно, устройство не увеличивает этот показатель, чтобы не перегружать сеть. Здесь, как и на канальном уровне, для обеспечения правильного функционирования всей сетевой инфраструктуры большое значение имеет взаимопонимание.

Таким образом, сущность вышеописанной стратегии заключается в том, что при высокоскоростном соединении и низкой загруженности сети данные будут отправляться быстро, в противном же случае передача данных замедлится до уровня соответствия ограничениям сетевых соединений между исходным и конечным устройствами.

2.4. Прикладной уровень

Канальный, сетевой и транспортный уровни работают слаженно, что позволяет быстро и надежно перемещать данные между двумя устройствами в комплексе сетей. Исходя из этого вытекает вопрос о том, какие сетевые приложения создаются для использования таких сетевых соединений.

В середине 1980-х годов, во времена появления первого интернета, сетевые приложения разрешали пользователям входить в систему на удаленных устройствах, передавать файлы, отправлять почту и даже пользоваться текстовыми чатами в режиме реального времени.

В начале 1990-х годов, когда интернет стал доступен большему количеству людей и устройства научились без труда обрабатывать изображения, учеными крупнейшей в мире лаборатории физики высоких энергий *ЦЕРН* (European Organization for Nuclear Research – CERN) было разработано приложение World Wide Web (Всемирная паутина). Тогда оно было ориентировано на чтение и редактирование сетевых гипертекстовых документов с изображениями. Сегодня же Всемирная паутина представляет собой обширное сетевое приложение, используемое повсеместно. Однако наряду с ним также используются и другие интернет-приложения.

Каждое приложение имеет две стороны. Одна из них называется серверной. Она функционирует на конечном устройстве и ожидает входящих сетевых подключений. Другая сторона называется клиентской и запускается на исходном устройстве. Браузеры, например

Firefox, Chrome или Internet Explorer, являются веб-клиентами. Они устанавливают соединения с веб-серверами и отображают страницы и документы, хранящиеся на этих веб-серверах. *Унифицированные указатели ресурсов* (Uniform Resource Locator – URL), которые вы можете видеть в адресной строке браузера, являются адресами веб-серверов, с которыми работает ваш клиент.

При разработке серверной и клиентской сторон приложения необходимо определить протокол приложения, с помощью которого в дальнейшем стороны будут обмениваться сообщениями по сети. Для каждого приложения используется свой протокол, каждый из которых опирается на задачи конкретного приложения. Несколько позже мы рассмотрим некоторые из таких протоколов.

2.5. Расположение слоев

Обычно мы представляем TCP/IP-модель в виде четырех уровней (канального, сетевого, транспортного и прикладного), расположенных в виде слоев, где прикладной уровень – верхний, а канальный – нижний. Причина, по которой их изображают именно таким образом, заключается в том, что в процессе сетевой коммуникации каждый слой взаимодействует с уровнями над и под собой.

Все четыре уровня работают как на исходном устройстве, где пользователь запускает клиентское приложение (например, браузер), так и на конечном устройстве, где запускается сервер приложений. Конечный пользователь взаимодействует с приложениями, составляющими верхний уровень модели, а на нижнем функционирует Wi-Fi, сотовое и проводное соединение между исходным устройством и остальной частью интернета.

Маршрутизаторы имеют своей целью лишь перемещение пакетов данных к месту назначения, поэтому они не взаимодействуют с транспортным и прикладным уровнями. Маршрутизаторы работают на сетевом и канальном уровнях. Для транспортировки и обеспечения переходов пакетов им необходимы только адреса источника и назначения на сетевом уровне. Транспортный и прикладной уровни вступают в игру только после того, как сетевой уровень доставит пакеты на конечное устройство.

Если вы хотите создать собственное сетевое приложение, вы, скорее всего, будете работать только с транспортным уровнем и не коснетесь сетевого и канального уровней. Безусловно, они необходимы для работы транспортного уровня, но при написании программ детали нижнего уровня не так важны. Многоуровневая сетевая модель

упрощает написание сетевых приложений, поскольку появляется возможность упускать многие сложные детали перемещения данных с одного устройства на другой.

Далее мы поговорим о вышеописанных уровнях более подробно.

2.6. Глоссарий

Клиент – в сетевом приложении является запрашивающей или инициирующей соединение стороной.

Волоконно-оптический кабель – технология передачи данных, которая кодирует данные с помощью света, передаваемого по очень длинной нити из тонкого стекла или пластика. Оптоволоконные соединения работают быстро и могут покрывать очень большие расстояния.

Относительный адрес – относительное положение пакета в сообщении или потоке данных.

Сервер – в сетевом приложении отвечает на запросы или ожидает входящих подключений.

Размер группы сетевых пакетов – объем данных, который исходное устройство может отправить до ожидания подтверждения.

2.7. Контрольные вопросы

Вы можете пройти этот тест онлайн по адресу **http://www.net-intro.com/quiz/**.

1. Почему инженеры организуют подход к решению большой и сложной задачи путем создания модели?
   А. Потому что это позволяет им построить что-то маленькое и протестировать это в аэродинамической трубе.
   Б. Потому что чем дольше они будут что-либо обсуждать, тем позже им придется приступить к самой тяжелой части работы.
   В. Потому что они могут разбить задачу на набор более мелких задач, которые могут быть решены независимо друг от друга.
   Г. Потому что это помогает в разработке маркетинговых материалов.

2. Какой из слоев находится наверху сетевой модели и используется сетями TCP/IP?
   А. Прикладной.
   Б. Транспортный.

В. Сетевой.
Г. Канальный.

3. Какой из уровней заботится о получении пакета данных через одно физическое соединение?
   А. Прикладной.
   Б. Транспортный.
   В. Сетевой.
   Г. Канальный.

4. Что означает CSMA/CD?
   А. Множественный доступ с прослушиванием несущей и обнаружением конфликтов.
   Б. Доступ к среде с контролем конфликтов с непрерывным направлением.
   В. Коррелированное размещение космических носителей с постоянным разделением.
   Г. Разделение каналов с постоянным состоянием на несколько адресов.

5. В чем заключается задача сетевого уровня?
   А. Убедиться, что данные не будут потеряны во время маршрута.
   Б. Получить пакет данных, перемещенный по нескольким сетям от источника к месту назначения.
   В. Убедиться, что только зарегистрированные пользователи могут использовать интернет.
   Г. Убедиться, что Wi-Fi распределяется между несколькими устройствами равномерно.

6. Что, помимо адресов данных, источника и получателя, требуется для того, чтобы убедиться, что сообщение может быть восстановлено, когда оно достигнет места назначения?
   А. Относительный адрес.
   Б. Место для отправки данных, если устройство назначения не работает .
   В. Сжатая и несжатая версия данных в пакете.
   Г. GPS-координаты конечного устройства.

7. Что такое размер группы сетевых пакетов?
   А. Сумма длины и ширины пакета.
   Б. Максимальный размер одного пакета.

В. Максимальное количество пакетов, которые могут входить в состав сообщения.
Г. Максимальный объем данных, который устройство может отправить до получения подтверждения.

8. Где в типичном сетевом клиент-серверном приложении запускается клиентское приложение?
   А. На портативном, стационарном или мобильном устройстве.
   Б. На точке беспроводного доступа.
   В. На ближайшем маршрутизаторе.
   Г. В оптоволоконном кабеле.

9. Что означает URL?
   А. Универсальная маршрутизация (Universal Routing Linkage).
   Б. Единая логика повторной передачи (Uniform Retransmission Logic).
   В. Унифицированный указатель ресурса (Uniform Resource Locator).
   Г. Единый список восстановления (Unified Recovery List).

Глава 3

Канальный уровень

Канальный уровень – это нижний уровень интернет-архитектуры. Он называется нижним, поскольку расположен наиболее близко к физическому уровню сети. На канальном уровне данные передаются с помощью проводов, оптоволоконных кабелей или радиосигналов. Особенность этого уровня заключается в том, что по нему данные передаются только в одном сегменте локальной сети. К канальному уровню относятся проводной Ethernet, Wi-Fi и сеть сотовой связи. Все эти технологии позволяют передавать данные в пределах километра. Волоконно-оптические линии, в том числе подводные, позволяют передавать данные на расстояние до тысяч километров. Также передавать данные на большие расстояния возможно с помощью спутниковых линий передачи данных.

Рис. 3.1. Канальный уровень

Независимо от расстояния на данном уровне данные передаются по одному каналу. Так, на пути к конечному устройству, как правило, требуется пересылка пакетов по нескольким каналам. В этом разделе мы подробно рассмотрим, как работает одна из наиболее распростра-

ненных технологий канального уровня – Wi-Fi. Данная технология является отличным примером для рассмотрения некоторых проблем канального уровня, требующих решения.

3.1. Беспроводная передача данных

Когда вы подключаетесь к интернету с помощью Wi-Fi с компьютера или смартфона, отправка и прием данных на вашем устройстве осуществляется посредством небольшого радиомодуля, работающего на низкой мощности. Он способен передавать данные лишь на небольшие расстояния (около 300 м), поэтому на следующем этапе пакеты ваших данных отправляются на маршрутизатор, который может находиться, например, у вас дома. Далее, по каналу передачи данных маршрутизатор пересылает данные в интернет. Такие маршрутизаторы называют базовыми станциями или шлюзами.

Устройства с включенными радиомодулями, находящиеся достаточно близко к базовой станции, получают все пакеты, передаваемые этой станцией, независимо от того, на какое устройство назначена отправка пакета. Помимо этого, они могут принимать все пакеты, отправляемые соседними устройствами. Таким образом, вашему устройству необходим способ, который поможет разграничить свои и чужие пакеты.

Часто способность устройств принимать множество пакетов в зоне досягаемости привлекает мошенников. Так они могут незаконно получать важные данные, такие как номера банковских счетов или пароли к онлайн-сервисам. В следующем разделе мы вернемся к вопросу о безопасности личных данных.

На производстве каждому Wi-Fi-радиомодулю присваивается уникальный серийный номер. Таким образом, каждое устройство, поддерживающее технологию Wi-Fi, имеет собственный серийный номер, как и радиомодуль в шлюзе. Серийный номер радиомодуля Wi-Fi можно найти в настройках устройства. Обычно он выглядит следующим образом:

```
0f:2a:b3:1f:b3:1a
```

Такой 48-битный серийный номер часто называют *адресом управления доступом к среде передачи данных* (Media Access Control), или MAC-адресом. MAC-адрес можно сравнить с адресом на почтовом конверте. Каждый пакет (радиоконверт), отправленный через сеть Wi-Fi, содержит адреса исходного и конечного устройств, благодаря

чему устройства с легкостью могут определить предназначенные для них пакеты.

При подключении к Wi-Fi, устройству необходимо выяснить, какой из MAC-адресов можно использовать для отправки пакетов на маршрутизатор. Если вы физически переместите ваше устройство с одного место на другое, оно будет взаимодействовать с разными шлюзами, каждый из которых имеет свой серийный номер. Поэтому, когда вы впервые подключаетесь к новой сети Wi-Fi, устройству необходимо обнаружить ее MAC-адрес.

Для того чтобы узнать MAC-адрес, устройство отправляет специальное сообщение на широковещательный адрес. Фактически оно как бы задает вопрос: «Кто отвечает за эту сеть Wi-Fi?» В этом сообщении устройство указывает свой серийный номер в качестве адреса отправителя («от»), а в качестве адреса получателя («кому») он указывает широковещательный адрес. Таким образом, устройство может узнать, есть ли шлюз в этой Wi-Fi-сети.

```
От: 0f:2a:b3:1f:b3:1a
Кому: ff:ff:ff:ff:ff:ff
Сообщение: Есть ли MAC-шлюз в этой сети?
```

Если в сети есть шлюз, он отправляет ответное сообщение, содержащее его серийный номер.

```
От: .98:2f:4e:78:c1:b4
Кому: 0f:2a:b3:1f:b3:1a
Сообщение: Я шлюз
Добро пожаловать в мою сеть
```

В случае, если ответа не поступает, ваше устройство ждет несколько секунд, а затем решает, что в этой сети нет шлюза. При отсутствии шлюза на устройстве может отображаться другой значок Wi-Fi или же он может не отображаться вообще. Бывают случаи, когда в одной сети может быть сразу несколько шлюзов. Однако пока что мы упустим этот момент, поскольку данное явление не распространено и к тому же довольно сложно для понимания.

Когда ваше устройство получает сообщение с MAC-адресом шлюза, оно может использовать этот адрес для отправки пакетов в интернет. На этом этапе всем пакетам вашего устройства присваивается серийный номер конечного устройства. Поэтому следует использовать широковещательный адрес как можно меньше, так как каждое устройство, подключенное к сети Wi-Fi, получает и обрабатывает лю-

бые сообщения, отправленные на широковещательный адрес, чтобы убедиться, что сообщения предназначены не для них.

3.2. Координация обмена данными

Многие устройства используют одни и те же радиочастоты, именно поэтому крайне важно координировать то, как они отправляют данные. К примеру, если в комнате находится несколько человек и все они одновременно начинают говорить, никто из собеседников не сможет услышать и понять друг друга. То же самое происходит, когда несколько радиомодулей Wi-Fi одновременно передают данные на одной и той же частоте. Именно поэтому необходим способ, позволяющий координировать работу всех радиомодулей. В этом разделе мы рассмотрим основы технических подходов к предотвращению потери данных ввиду возникновения конфликтов при передаче.

Первый подход называется «контроль несущей». Данный подход заключается в том, что сначала канал передачи данных прослушивается на наличие осуществляемой в данный момент передачи. Если оказывается, что канал занят, устройство ожидает завершения активного процесса. Можно предположить, что ожидание занимает долгое время, но, поскольку все сообщения разбиваются на пакеты, обычно устройству достаточно дождаться только завершения отправки пакета другим устройством, после чего оно может приступать к отправке своих пакетов.

Если при прослушивании канала передачи данных Wi-Fi-радиомодуль устройства не обнаруживает активных процессов, оно может начать передачу. Но что, если Wi-Fi-радиомодуль другого устройства, также не обнаружив активной передачи, начнет передавать свои данные? Если несколько радиомодулей начинают передачу одновременно, вероятнее всего, данные повредятся, и оба пакета будут потеряны. Поэтому радиомодулю необходимо продолжать прослушивание канала при отправке пакетов. Так он убедится что сможет принимать собственные данные. Если же радиомодуль не может принять собственные данные, он предполагает, что произошел конфликт (это называется обнаружением конфликтов), и прекращает передачу.

То же самое происходит, например, в комнате, где несколько людей пытаются вести дискуссию, и тогда часто бывает, что два человека начинают говорить одновременно между собой. Но они сразу замолкают, уступая друг другу возможность высказаться, и в итоге молчат оба. Как же в такой ситуации разговор возобновляется? Обычно это происходит так: после долгой паузы оба человека снова начинают

разговаривать одновременно. Так может продолжаться несколько раз, поэтому зачастую подобные ситуации выглядят крайне комично.

Радиомодули, в отличие от людей, способны решить эту проблему намного быстрее. Когда радиомодули обнаруживают конфликт данных или искажение передачи, перед повторной попыткой передачи они рассчитывают случайное время ожидания. Правила расчета случайного ожидания устанавливаются таким образом, чтобы оно разнилось у каждого конфликтующего устройства.

Формальное название для процесса, включающего в себя прослушивание, передачу, прослушивание, ожидание и повторную попытку передачи (в случае необходимости), – «множественный доступ с контролем несущей и обнаружением конфликтов», или CSMA/CD.

На первый взгляд такая технология не внушает доверия. Может показаться странным что вся ее сущность заключается лишь в том, что при возникновении конфликта устройство просто делает попытку передачи, а затем через какое-то время делает ее еще раз. Однако на практике эта технология применяется крайне успешно. Существует целая категория технологий канального уровня, работающая по этому шаблону (прослушивание, передача, прослушивание и повторная передача (при необходимости)). В нее входят проводной Ethernet, данные сотовой связи и даже служба коротких сообщений (SMS/текстовые сообщения).

3.3. Координация в других технологиях канального уровня

В структуру технологий канального уровня, имеющих большое количество передающих станций (ввиду чего им приходится работать с почти 100%-й эффективностью в течение длительных периодов времени), заложен иной подход. Ключевым понятием данного подхода является «токен». Токен указывает, когда каждой станции предоставляется возможность передачи данных. Станции не могут начать передачу, если у них нет токена. Таким образом, от них требуется только дождаться своей очереди.

Когда станция получает токен, она отправляет имеющийся пакет. После отправки станция передает токен далее и затем ждет его возвращения. Если ни одна из станций не имеет данных для отправки, токен просто перемещается с одного устройства на другое.

Такой подход можно сравнить с ситуацией, когда группа людей передает мяч по кругу, и при этом говорить может только тот, кто дер-

жит мяч в руках. Таким образом, они общаются не перебивая друг друга. Когда один человек получает мяч, он высказывает свою мысль (передает пакет слов), а затем передает мяч дальше.

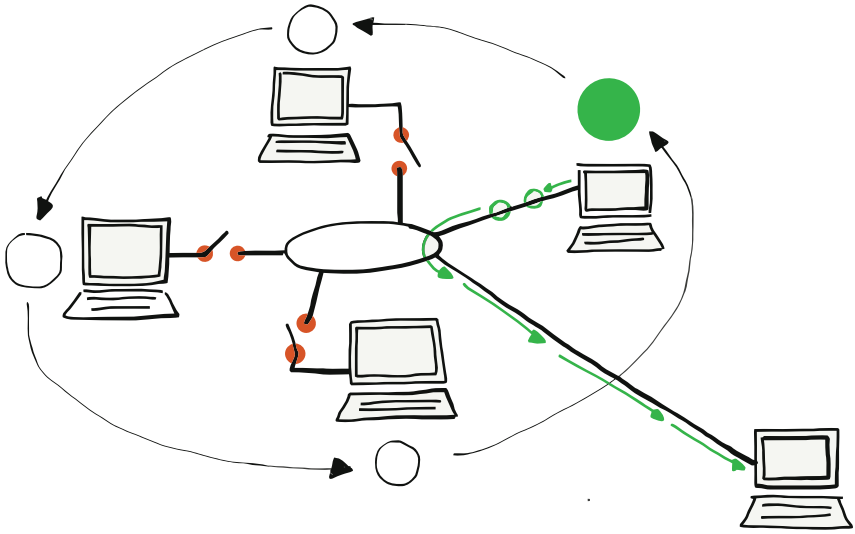

Рис. 3.2. Обмен данными с помощью токена

Подход CSMA/CD применим при отсутствии данных или при отправке данных низкого или среднего уровней. В сети, где используется технология токена, перед отправкой данных устройству необходимо подождать даже в том случае, если активной передачи в настоящий момент не происходит. Когда устройство завершит отправку пакетов, ему необходимо будет дождаться, пока токен вернется, и только потом оно сможет отправить следующий пакет. Если в сети находится лишь одна станция, желающая отправить данные, ей все равно придется ожидать прохождения токена через все другие станции.

Вышеописанный подход лучше всего подходит при использовании таких сред, как спутниковая связь или подводные оптоволоконные линии, где обнаружение конфликта может занимать слишком много времени или быть слишком затратным. CSMA/CD (метод «прослушай-попробуй») лучше всего подходит для не слишком затратных сред, покрывающих меньшие расстояния и имеющих множество совместно работающих станций, отправляющих данные в виде коротких пакетов. Именно поэтому Wi-Fi (и CSMA/CD) особенно эффективен для обеспечения доступа к сети в кафе, домах, квартирах или школах.

3.4. Заключение

Итак, мы рассмотрели нижний уровень интернет-архитектуры. Однако лишь вкратце. Безусловно, на этом уровне существует множество других важных деталей, таких как расстояние подключения, напряжение, частота, скорость и многие другие. Каждая из этих деталей требует углубленного изучения.

Ключевым преимуществом многоуровневой архитектуры является то, что инженеры, создающие технологии канального уровня, могут игнорировать проблемы, которые решаются на уровнях выше. Так, они могут сосредоточиться на создании оптимального решения для передачи данных на канальном уровне. Современные технологии канального уровня, такие как Wi-Fi, спутниковые и кабельные модемы, Ethernet и сотовая связь представляют собой крайне развитые системы. Данные перемещаются быстро и без разрывов, благодаря чему пользователю не нужно беспокоиться о том, что происходит на этом уровне. Он просто работает.

3.5. Глоссарий

Базовая станция – название первого маршрутизатора, который обрабатывает пакеты по мере их передачи в интернет.

Широковещательная передача – отправка пакета таким образом, чтобы ее смогли получить все станции, подключенные к локальной сети.

Шлюз – маршрутизатор, подключающий локальную сеть к более широкой сети (например, интернет). С помощью шлюзов устройства могут отправлять данные за пределы локальной сети.

MAC-адрес – адрес, назначаемый сетевому оборудованию при производстве.

Токен – метод, позволяющий нескольким устройствам совместно использовать один и тот же физический носитель без риска возникновения конфликтов. Перед отправкой данных каждому устройству необходимо дождаться получения токена.

3.6. Контрольные вопросы

Вы можете пройти этот тест онлайн по адресу **http://www.net-intro.com/quiz/**.

1. Куда ваше устройство отправляет пакеты при подключении к интернету через сеть Wi-Fi?

А. Шлюз.
Б. Спутник.
В. Вышка сотовой связи.
Г. Интернет-центр.

2. Каким образом назначается канал передачи / физический адрес для сетевого устройства?
   А. Вышкой сотовой связи.
   Б. «Администрацией адресного пространства Интернет» (Internet Assignment Numbers Authority – IANA).
   В. Производителем оборудования связи.
   Г. Правительством.

3. Какой из перечисленных вариантов является адресом канала?
   А. 0f:2a:b3:1f:b3:1a.
   Б. 192.168.3.14.
   В. www.khanacademy.com.
   Г. @drchuck.

4. Как устройство находит шлюз в сети Wi-Fi?
   А. Оно имеет адрес шлюза, установленный производителем.
   Б. Оно передает широковещательный запрос адреса шлюза.
   В. Оно многократно отправляет сообщение на все возможные адреса шлюза, пока не найдет его.
   Г. Пользователь должен вводить адрес шлюза вручную.

5. Что в первую очередь делает устройство перед началом отправки данных по Wi-Fi?
   А. Осуществляет прослушивание канала на предмет активной передачи.
   Б. Просто начинает отправлять данные.
   В. Отправляет на шлюз сообщение с запросом разрешения на передачу.
   Г. Ждет своей очереди.

6. Что делает передающая станция, подключенная к Wi-Fi, когда она пытается отправить данные, но обнаруживает, что произошел конфликт?
   А. Продолжает отправлять сообщение, чтобы хотя бы часть сообщения передалась.
   Б. Ожидает, пока шлюз не сообщит, что конфликт разрешен.
   В. Немедленно перезапускает передачу сообщения в начале.

Г. Останавливает передачу и делает паузу перед повторной попыткой.

7. Что в первую очередь делает устройство перед началом передачи данных с помощью токена?
   А. Осуществляет прослушивание канала на предмет активной передачи.
   Б. Просто начинает отправлять данные.
   В. Отправляет на шлюз сообщение с запросом разрешения на передачу.
   Г. Ждет своей очереди.

Глава 4

Сетевой уровень (IP)

Теперь, когда мы узнали, как передаются данные по одному каналу связи, пришло время выяснить, каким образом они передаются на большие расстояния, в пределах одной страны или даже по всему миру. По мере прохождения своего пути от одного устройства к другому данные осуществляют несколько переходов через несколько сетей. Такой путь можно сравнить с путешествием в далекую страну. Так, сначала путешественник идет от дома до автобусной остановки, затем пересаживается на поезд до города, далее садится на другой поезд до аэропорта, затем на самолете он прилетает в другой аэропорт и едет на такси до большого города, где затем садится на поезд, чтобы доехать до маленького городка и после этого наконец идет от остановки до своего отеля. Как и путешественники, пакеты могут использовать различные виды «транспорта». Для пакета, отправляющегося в «путешествие», под видами транспорта подразумеваются различные технологии канального уровня, такие как Wi-Fi, Ethernet, оптоволоконные кабели и спутниковая связь.

Во время путешествия человек (и пакет) может пользоваться общественным транспортом. В одном автобусе, поезде или самолете с вами могут находиться сотни других людей. Однако ввиду решений, которые вы принимаете после каждого своего перехода, маршрут вашей поездки отличается от маршрута любого из них. Например, когда вы прибываете на вокзал, то можете выйти из одного поезда, а затем пересесть на другой отбывающий поезд. Путешественники, имеющие как разные отправные точки, так и пункты назначения, как правило, принимают разные решения. По мере принятия таких решений вы продвигаетесь по маршруту (делаете переходы), который приведет вас к месту назначения.

По мере транспортировки пакет также проходит ряд станций, на каждой из которых принимается решение о том, по какому выходному каналу он будет перенаправлен далее. Такими станциями вы-

ступают маршрутизаторы. Маршрутизаторы (как и железнодорожные станции) имеют множество входящих и исходящих каналов. Некоторые каналы могут быть оптоволоконными, другие – спутниковыми, а третьи – беспроводными. Задача маршрутизатора – убедиться, что пакеты, проходящие через маршрутизатор, попадают на нужный уровень исходящих каналов. Обычно по мере прохождения своего пути пакет проходит от 5 до 20 маршрутизаторов.

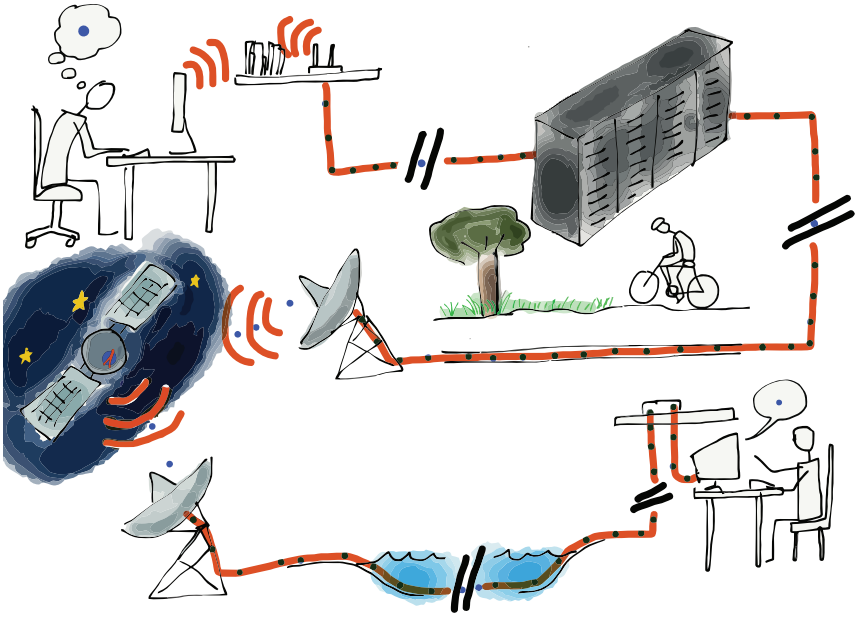

Рис. 4.1. Путешествие пакетов

Если на железнодорожных станциях для пассажиров есть специальный экран, где перечислены поезда и маршруты, то маршрутизаторы ориентируются по адресу назначения, который содержится в пакете. Так они решают, какой исходящий канал должен принять этот пакет. Подобный подход можно сравнить с тем, как если бы служащий вокзала встречал каждого человека, выходящего из прибывающего поезда, и провожал до другого поезда.

Маршрутизатор может быстро определить исходящую ссылку, так как каждый отдельный пакет содержит информацию о конечном адресе. Такой адрес называют адресом интернет-протокола (Internet Protocol Address), или сокращенно – IP-адресом. С помощью IP-адреса

можно значительно повысить эффективность работы маршрутизатора по пересылке пакетов.

4.1. IP-адреса (Internet Protocol)

В предыдущем разделе мы говорили об адресах канального уровня и выяснили, что такие адреса назначаются на этапе производства оборудования и не меняются на протяжении всего срока службы устройства. Однако адреса канального уровня не применимы при маршрутизации пакетов через несколько сетей, поскольку связь между адресом канального уровня и местом, где устройство подключается к сети, отсутствует. Например, портативные компьютеры и сотовые телефоны постоянно перемещаются в пространстве, поэтому в таком случае системе необходимо будет отслеживать перемещения каждого отдельного устройства. Таким образом, поскольку в сети существуют миллиарды устройств, использование адресов канального уровня при маршрутизации представляется нецелесообразным.

Рис. 4.2. Сетевой уровень

Более эффективным инструментом для маршрутизации сообщений выступает IP-адрес, который назначается устройству в соответствии с его местоположением. На сегодняшний день существует две версии IP-адресов. Старые (классические) адреса IPv4 состоят из четырех чисел, разделенных точками, и выглядят следующим образом:

212.78.1.25.

Максимальное число в IP адресе – 255. Сегодня существует настолько много устройств, подключенных к интернету, что уникальные

адреса IPv4 для них заканчиваются. IPv6-адреса длиннее и выглядят так:

```
2001:0db8:85a3:0042:1000:8a2e:0370:7334.
```

В этом разделе мы сосредоточимся на классических адресах IPv4, однако стоит отметить, что все представленные здесь идеи в равной степени применимы как к адресам IPv4, так и к IPv6.

Самое важное, что можно сказать об IP-адресах, – это то, что они делятся на две части[1]. Первая часть называется «адрес сети». Если мы разделим IPv4-адрес на две части, мы получим следующее:

```
Адрес сети: 212.78
Идентификатор хоста: 1.25.
```

Основная идея здесь состоит в том, что множество устройств может быть подключено к интернету через одно соединение. Например, весь кампус колледжа, школа или предприятие могут подключаться к сети, используя один или несколько определенных сетевых номеров. В приведенном выше примере 65 536 устройств может быть подключено к сети с использованием сетевого номера «212.78». Все устройства подключаются к интернету посредством одного соединения, поэтому все пакеты с IP-адресом 212.78.*.* перенаправляются в одно и то же место.

Благодаря такому подходу маршрутизаторам больше не нужно отслеживать миллиарды отдельных устройств. Вместо этого им требуется отслеживать около миллиона (или, возможно, меньше) сетевых номеров.

Таким образом, когда пакет прибывает в маршрутизатор и последнему необходимо определить дальнейший маршрут первого, маршрутизатору не требуется просматривать IP-адрес целиком. Для того чтобы определить оптимальный маршрут, ему достаточно взглянуть лишь на первую часть адреса.

4.2. Как маршрутизаторы определяют маршруты

Несмотря на то что описанный выше подход значительно сокращает количество конечных точек, которые должен отслеживать маршрутизатор, каждому из них все же требуется способ, позволяющий опреде-

[1] В представленном примере мы делим IP-адрес пополам, хотя обычно разбить его на «адрес сети» и «идентификатор хоста» можно и по-другому.

лить маршрут от самого маршрутизатора к каждому из сетевых номеров, с которыми он может столкнуться.

При первом подключении к интернету новый магистральный маршрутизатор не может знать все маршруты. Он может знать лишь несколько заранее сконфигурированных маршрутов, однако для оптимальной маршрутизации пакетов в дальнейшем ему необходимо отслеживать маршруты при обнаружении пакетов.

При обнаружении пакета маршрутизатор не сразу может определить оптимальный путь. Вместо этого он совершает запрос к соседним маршрутизаторам, которые уже сталкивались с данным номером сети. Они же в свою очередь отправляют ему необходимые данные. Иногда маршрутизаторы отправляют запросы «соседям» до тех пор, пока весь маршрут фактически не будет определен.

В самых простых случаях новый магистральный маршрутизатор может постепенно построить карту сетевых номеров для исходящих каналов, которая поможет ему правильно маршрутизировать пакеты на основе IP-адреса для каждого входящего пакета. Такая карта называется таблицей маршрутизации.

Когда интернет работает нормально, каждый маршрутизатор имеет относительно полную таблицу маршрутизации и редко встречается с незнакомыми сетевыми номерами. Как только маршрутизатор впервые определяет маршрут к новому сетевому номеру, ему уже не требуется повторно прокладывать маршрут для сетевого номера (разумеется, если что-то не изменится или не пойдет не так). Так, маршрутизатор выполняет поиск маршрута по первому пакету, а затем он может направить следующие пакеты на этот номер сети, просто используя информацию, уже содержащуюся в его таблицах маршрутизации.

4.3. Возможные проблемы и пути их решения

Разумеется, случается так, что в сети возникают проблемы. Однако маршрутизатор должен найти способ обойти их и выполнить свою функцию. Одной из самых распространенных проблем является сбой в исходящей линии. Причин у этой проблемы может быть масса, вплоть до того, что кто-то споткнулся о кабель и выдернул его из разъема. В таких случаях маршрутизаторы обычно уже располагают набором сетевых номеров, которые они хотели отправить по поврежденной линии. На удивление процесс восстановления при сбоях в исходящих линиях происходит достаточно просто. Сначала маршрутизатор отбрасывает все записи в своей таблице. Затем по мере поступления

новых пакетов, предназначенных для этих сетевых номеров, маршрутизатор заново выстраивает маршруты, на этот раз «советуясь» с соседними маршрутизаторами (кроме тех, с кем из-за разрыва связи он больше не может контактировать).

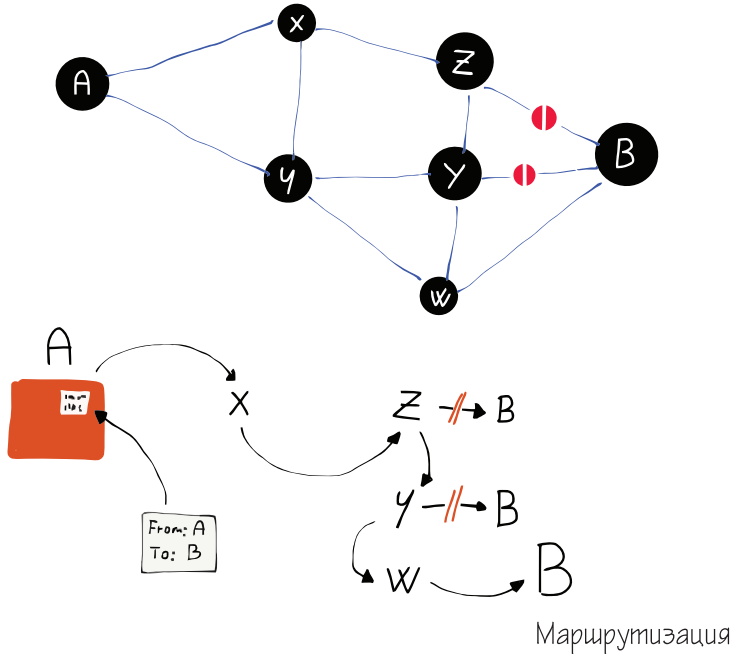

Рис. 4.3. Динамическая маршрутизация

На какое-то время работа маршрутизаторов замедляется ввиду перестраивания таблиц маршрутизации, выстраивающих новую конфигурацию сети. Через некоторое время стандартная скорость работы восстанавливается.

Вот почему так важно наличие как минимум двух независимых маршрутов от исходной сети к конечной. Если таковые имеются, мы называем такую сеть «сетью с двумя соединениями». Подобные сети могут легко восстанавливаться после сбоев в исходящих каналах. В местах, где сосредоточено множество сетевых подключений, например на восточном побережье США, в сети может произойти несколько сбоев исходящих линий, но при этом она все еще будет работать. Однако в некоторых частных случаях, например когда вы находитесь дома или в школе и имеете только одно соединение, если в нем возникает сбой, доступ к интернету полностью прекращается.

В какой-то момент поврежденная линия восстанавливается или вместо нее создается новая, и тогда маршрутизатор ищет способы использовать новые линии наиболее эффективно. Он заинтересован в совершенствовании собственных таблиц маршрутизации и в свободное от выполнения своих функций время ищет возможности улучшить таблицы. Во время пауз маршрутизатор запрашивает у соседа всю или часть его таблицы маршрутизации. Маршрутизатор просматривает таблицы соседей и, если он видит, что другой маршрутизатор построил более оптимальный маршрут к какому-либо сетевому номеру, он обновляет свою таблицу.

Благодаря всем вышеописанным процессам маршрутизаторы учатся быстрее реагировать на сбои и перенаправлять пакеты с поврежденных или медленных линий на нормально функционирующие и/или более быстрые. Маршрутизаторы постоянно взаимодействую между собой, что позволяет им совершенствовать свои таблицы маршрутизации. Несмотря на то что никто не может подсказать маршрутизаторам универсальный «наилучший маршрут», они почти всегда могут определить самый быстрый путь от источника к пункту назначения. Также они легко справляются с обнаружением пакетов, передающихся по медленным или временно перегруженным линиям и их динамической маршрутизацией.

Бывают случаи, что маршруты пакетов могут меняться по ходу их прохождения от исходного к конечному устройству. Также вы можете отправить один пакет, а затем сразу же второй, и в итоге второй пакет прибудет в пункт назначения раньше первого. Однако на IP-уровне порядок пакетов не отслеживается; здесь решается множество других важных задач.

Отправку пакетов с IP-адресами исходного и конечного устройств можно сравнить с отправкой писем по почте. Каждый пакет проходит через систему и в конце концов достигает места назначения.

4.4. Определение маршрута

Ни один из элементов интернет-архитектуры заведомо не имеет информации о маршруте, по которому ваши пакеты будут идти от исходного устройства к конечному. Всего маршрута, по которому они будут проходить, не знают даже маршрутизаторы, непосредственно участвующие в пересылке пакетов. Задача маршрутизаторов состоит лишь в том, чтобы определять маршруты переходов пакетов, которые приближают их к пункту назначения.

Оказывается, что на большинстве устройств есть инструмент сетевой диагностики под названием traceroute (или tracert, в зависимости от операционной системы), который позволяет отслеживать маршруты следования пакетов. Однако, учитывая, что маршруты пакетов могут меняться, отслеживание с помощью данного инструмента является не более чем «неплохим предположением» относительно фактических маршрутов.

На самом деле traceroute вообще не «отслеживает» пакет. Данная утилита работает на базе маршрутизируемого сетевого протокола (IP), благодаря которому пакеты не «застревают» в сети. Прежде чем мы перейдем непосредственно к traceroute, следует кратко рассмотреть, как пакет может «застрять» в сети и как IP решает эту проблему.

Помните, что информация о любом отдельном маршруте является лишь приблизительной оценкой оптимального исходящего канала для определенного номера сети. Так, маршрутизаторы не имеют информации о деятельности друг друга. Но что, если бы у нас было три маршрутизатора, имеющих записи в таблицах маршрутизации и образующих бесконечную петлю?

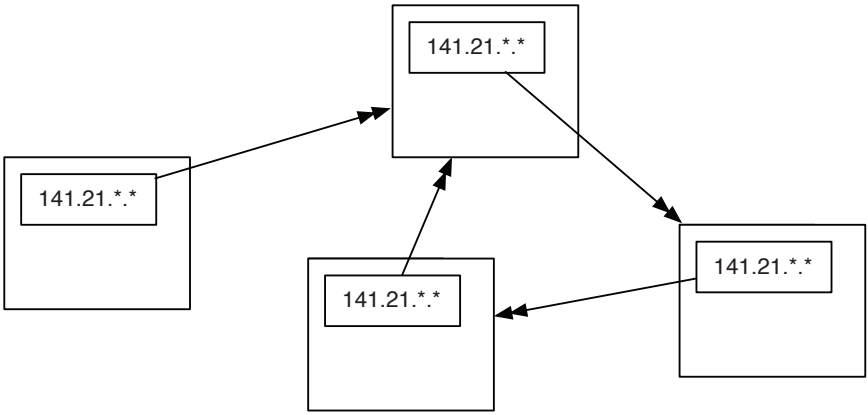

Рис. 4.4. «Маршрутный вихрь»

Каждый из маршрутизаторов считает, что знает наилучший исходящий канал для IP-адресов, начинающихся с «212.78». Однако в случае, представленном на рисунке, ввиду произошедшей путаницы

между маршрутизаторами образуется своего рода маршрутная петля. Если пакет с префиксом «212.78» попадет в один из этих маршрутизаторов, он будет бесконечно перенаправляться по кругу из трех каналов, не имея выхода. По мере поступления новых пакетов с тем же префиксом они просто добавятся в бесконечный «маршрутный вихрь». Каналы довольно быстро заполнятся циклическим трафиком, а маршрутизаторы – пакетами, ожидающими отправки, и в итоге все три маршрутизатора выйдут из строя. Такая проблема намного серьезнее, чем физический разрыв кабеля, поскольку она может привести к сбою в работе сразу нескольких маршрутизаторов.

Чтобы решить эту проблему, разработчики интернет-протокола добавили к каждому пакету значение, которое называется *временем жизни данных* (Time To Live – TTL). Минимальное TTL, как правило, начинается от 30. Каждый раз, когда IP-пакет совершает переход, маршрутизатор вычитает одну единицу из значения TTL. Таким образом, если пакету требуется 15 переходов для прохождения маршрута, то в конечной точке он появится с TTL 15.

Теперь давайте рассмотрим, как работает TTL при образовании маршрутной петли (или «маршрутного вихря») в определенном сетевом номере. В результате постоянного прохождения пакета через одни и те же каналы его значение TTL рано или поздно достигает нуля. Когда значение TTL достигает нуля, маршрутизатор понимает, что что-то пошло не так, и отбрасывает пакет. Такой подход гарантирует, что образование маршрутных петель не спровоцирует глобальных сбоев в сети.

Данный подход является отличным решением. Чтобы обнаружить маршрутные петли и выйти из них, мы просто вводим число, вычитаем по единице из этого числа при каждом переходе, а когда значение становится равным нулю, отбрасываем пакет.

Также, когда маршрутизатор отбрасывает пакет, он отправляет любезное уведомление, что-то вроде: «Извините, мне пришлось выбросить ваш пакет». В сообщении указан IP-адрес маршрутизатора, который отбросил пакет.

Маршрутные петли на самом деле встречаются довольно редко. Однако описанное выше уведомление может быть полезно при определении приблизительного маршрута пакета. Утилита traceroute отправляет пакеты таким способом, чтобы маршрутизаторы, через которые проходят ваши пакеты, отправляли им такие уведомления.

Сначала traceroute отправляет пакет с TTL, равным 1. Пакет попадает на первый маршрутизатор и сразу отбрасывается, а ваше устройство получает соответствующее уведомление. Затем traceroute отправляет пакет с TTL, равным 2. Пакет проходит через первый маршрутизатор и отбрасывается вторым, который снова отправляет вам уведомление. Затем traceroute отправляет пакет с TTL, равным 3. Так, с каждым разом утилита увеличивает значение TTL, пока пакет не дойдет до места назначения.

Именно так traceroute определяет приблизительный маршрут ваших пакетов. Например, чтобы добраться из Анн-Арбора, штат Мичиган, в Пало-Альто, штат Калифорния, пакетам потребовалось 14 переходов. Они прошли через Канзас, Техас, Лос-Анджелес и Окленд. Возможно, в сравнении с маршрутом машины или поезда, это не самый быстрый путь. Однако именно в этом случае данный маршрут был наилучшим для конкретных пакетов.

На рис. 4.5 вы можете увидеть, сколько времени потребовалось пакетам, чтобы добраться от исходного устройства до каждого маршрутизатора, а также от исходного устройства к конечному. Миллисекунда (мс) – это 1/1000 доля секунды. Таким образом, 77,317 мс составляет чуть меньше десятой доли секунды. Из этого следует вывод, что скорость данной сети довольно высока.

```
traceroute www.stanford.edu
traceroute to www5.stanford.edu (171.67.20.37), 64 hops max, 40 byte packets
 1  141.211.203.252 (141.211.203.252)  1.390 ms  0.534 ms  0.490 ms
 2  v-bin-seb.r-bin-seb.umnet.umich.edu (192.122.183.61)  0.591 ms  0.558 ms  0.570 ms
 3  v-bin-seb-i2-aa.merit-aa2.umnet.umich.edu (192.12.80.33)  6.610 ms  6.545 ms  6.654 ms
 4  192.122.183.30 (192.122.183.30)  7.919 ms  7.209 ms  7.122 ms
 5  so-4-3-0.0.rtr.kans.net.internet2.edu (64.57.28.36)  17.672 ms  17.836 ms  17.673 ms
 6  so-0-1-0.0.rtr.hous.net.internet2.edu (64.57.28.57)  31.800 ms  41.967 ms  31.787 ms
 7  so-3-0-0.0.rtr.losa.net.internet2.edu (64.57.28.44)  63.478 ms  63.704 ms  63.710 ms
 8  hpr-lax-hpr--i2-newnet.cenic.net (137.164.26.132)  63.093 ms  63.026 ms  63.384 ms
 9  svl-hpr--lax-hpr-10ge.cenic.net (137.164.25.13)  71.242 ms  71.542 ms  76.282 ms
10  oak-hpr--svl-hpr-10ge.cenic.net (137.164.25.9)  72.744 ms  72.243 ms  72.556 ms
11  hpr-stan-ge--oak-hpr.cenic.net (137.164.27.158)  73.763 ms  73.396 ms  73.665 ms
12  bbra-rtr.Stanford.EDU (171.64.1.134)  73.577 ms  73.682 ms  73.492 ms
13  * * *
14  www5.Stanford.EDU (171.67.20.37)  77.317 ms  77.128 ms  77.648 ms
```

Рис. 4.5. Маршрут из Мичигана в Стэнфорд

Иногда отслеживание маршрутов пакетов может занимать больше времени – до двух минут. Не все маршрутизаторы отправляют уведомление об отбрасывании пакета. В приведенном выше примере маршрутизатор на 13 этапе отбросил наш пакет, не сказав «Извините». Обычно traceroute несколько секунд ожидает такого уведомления и затем, если не получает его, отказывается от маршрутизатора и увеличивает значение TTL, чтобы пакет обошел его.

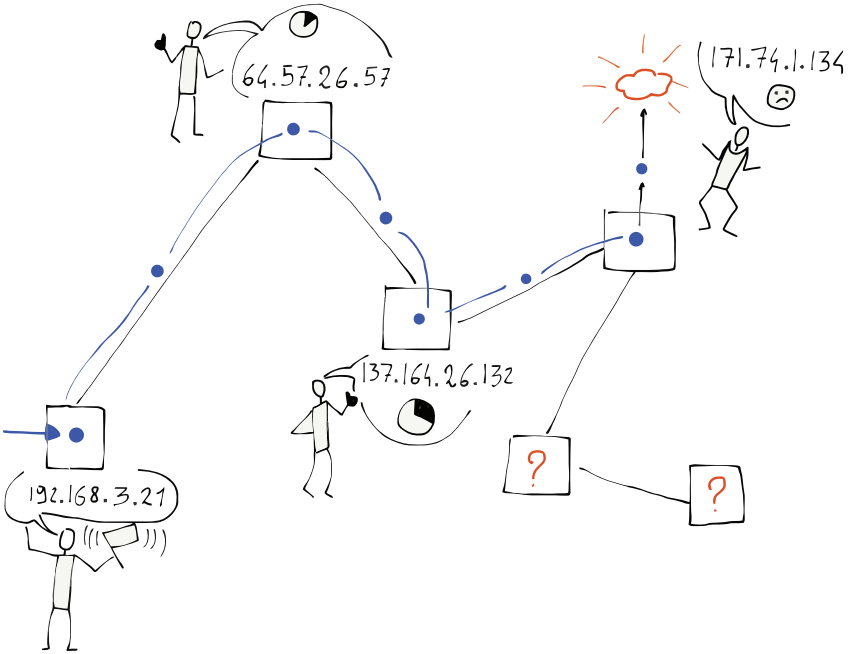

Рис. 4.6. Уведомления о потерянных пакетах

Если вы запустите traceroute для проводного соединения с подводным кабелем, вы увидите, насколько быстро данные перемещаются по подводным оптоволоконным линиям связи. На рис. 4.7 показан маршрут из Мичиганского университета в Пекинский университет в Китае.

```
$ traceroute www.pku.edu.cn
traceroute to www.pku.edu.cn (162.105.129.104), 64 hops max, 40 byte packets
 1  141.211.203.252 (141.211.203.252)  1.228 ms  0.584 ms  0.592 ms
 2  v-bin-seb.r-bin-seb.umnet.umich.edu (192.122.183.61)  0.604 ms  0.565 ms  0.466 ms
 3  v-bin-seb-i2-aa.merit-aa2.umnet.umich.edu (192.12.80.33)  7.511 ms  6.641 ms  6.588 ms
 4  192.122.183.30 (192.122.183.30)  12.078 ms  6.989 ms  7.619 ms
 5  192.31.99.133 (192.31.99.133)  7.666 ms  8.953 ms  17.861 ms
 6  192.31.99.170 (192.31.99.170)  59.275 ms  59.273 ms  59.108 ms
 7  134.75.108.209 (134.75.108.209)  173.614 ms  173.552 ms  173.333 ms
 8  134.75.107.10 (134.75.107.10)  256.760 ms 134.75.107.18 (134.75.107.18)  256.574 ms
 9  202.112.53.17 (202.112.53.17)  256.761 ms  256.801 ms  256.688 ms
10  202.112.61.157 (202.112.61.157)  257.416 ms  257.960 ms  257.747 ms
11  202.112.53.194 (202.112.53.194)  256.827 ms  257.068 ms  256.962 ms
12  202.112.41.202 (202.112.41.202)  256.800 ms  257.053 ms  256.933 ms
```

Рис. 4.7. Маршрут из Мичигана в Пекинский университет

Пакет доходит до подводного кабеля на этапах 7 и 8. Минимальное время изменяется от 1/10 до почти 1/4 с. Несмотря на то что 1/4 с

медленнее, чем 1/10 с, результат все равно впечатляет, если учесть, что за это время пакет совершает практически кругосветное путешествие.

Ядро нашей IP-сети удивительно. В большинстве случаев работа маршрутизаторов не влияет на скорость перемещения пакетов на дальних расстояниях. В следующем разделе мы рассмотрим, что происходит с IP-адресами при переходе пакетов из одного канала на другой.

4.5. Получение IP-адреса

На сегодняшний день все чаще люди пользуются портативными и мобильными устройствами. Ранее мы говорили о том, что для IP-уровня важнее отслеживать группы устройств, объединенных общим сетевым номером, чем отслеживать каждое устройство по отдельности. Но, поскольку сетевые номера указывают конкретное физическое подключение к сети, при каждом перемещении устройства в пространстве ему потребуется новый IP-адрес. Помните, что адрес канального уровня присваивается устройству при производстве и не меняется в течение всего срока службы. Если вы сначала поработаете со своего ноутбука, скажем, в кофейне, а затем вернетесь домой и будете работать с этого же устройства, но уже через домашний Wi-Fi, ноутбуку потребуется другой IP-адрес.

Способность устройства получать другой IP-адрес после перемещения из одной сети в другую использует протокол, называемый «Протокол динамической настройки узла» (Dynamic Host Configuration Protocol – DHCP). DHCP очень прост. В разделе «Канальный уровень», мы говорили о том, что на этом уровне первым делом устройство интересуется, есть ли в сети базовая станция, отправляя сообщение на специальный широковещательный адрес. После успешного подключения через базовую станцию оно отправляет другое сообщение на широковещательный адрес. На этот раз оно спрашивает: «Есть ли шлюз, подключенный к этой сети, который может соединить меня с интернетом? Если есть, сообщите мне свой IP-адрес и укажите, какой IP-адрес мне следует использовать в этой сети».

Когда маршрутизатор шлюза отвечает, вашему устройству предоставляется временный IP-адрес для этой сети (например, когда вы находитесь в кафе). После того как маршрутизатор перестает получать сообщения от вашего устройства, он решает, что вы ушли, и передает IP-адрес другому.

Если же два устройства оказываются в одной сети с одинаковым IP-адресом, в процессе повторного использования предоставленного IP-адреса происходит сбой. Возможно, вы уже видели сообщение типа «В этой сети уже есть компьютер с IP-адресом 192.168.0.5». Ваше устройство видит другое устройство с адресом канала, отличным от его собственного, при этом используя IP-адрес, который, по мнению вашего устройства, назначен ему.

Но в большинстве случаев протокол динамической настройки узла (DHCP) работает отлично. Вы открываете ноутбук, подключаетесь и уже через несколько секунд можете войти в интернет. Затем вы закрываете ноутбук и перемещаетесь в другое место, где устройству дается новый IP-адрес.

В некоторых операционных системах, когда устройство совершает DHCP-запрос при подключении к сети и не получает ответа, оно все равно решает присвоить себе IP-адрес. Очень часто такие адреса начинаются с «169. . . .». Если устройство присвоило один из этих IP-адресов, оно считает, что подключено к сети и имеет IP-адрес, но без шлюза у него нет возможности маршрутизировать пакеты через локальную сеть в интернет. В таких случаях все, что можно сделать, – подключить несколько устройств к одной локальной сети, найти друг друга и сыграть в сетевую игру. С такими IP-адресами ничего больше сделать не получится.

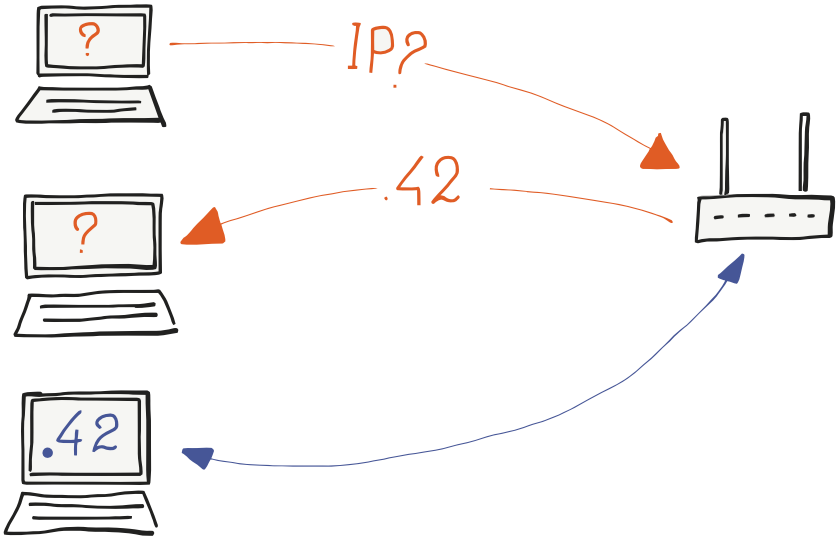

Рис.4.8. Получение IP-адреса через DHCP

4.6. Другой вид повторного использования адресов

Если вы знаете, как найти IP-адрес своего ноутбука, можно провести небольшой эксперимент и посмотреть, как ваше устройство получает разные IP-адреса в разных местоположениях. Составив список полученных по результатам эксперимента адресов, вы обнаружите, что многие из них начинаются на «192.168.». Может показаться, что таким образом нарушается правило, которое гласит, что сетевой номер (префикс IP-адреса) привязан к месту, где компьютер подключается к интернету. Однако к адресам, начинающимся с «192.168.», применяется другое правило (префикс «10.» тоже особенный).

Адреса, начинающиеся с «192.168.», называются немаршрутизируемыми. Это означает, что они не могут использоваться как реальные адреса, с помощью которых данные могут быть маршрутизированы через ядро сети. Немаршрутизируемые адреса можно использовать в локальной, но не в глобальной сети.

Тогда почему, когда ваше устройство получает адрес типа «192.168.0.5», оно продолжает нормально работать в глобальной сети? Это связано с тем, что ваш домашний маршрутизатор / шлюз / базовая станция пользуется механизмом «преобразования сетевых адресов» (Network Address Translation – NAT). Шлюз имеет один маршрутизируемый IP-адрес, который используется несколькими подключенными к нему рабочими станциями. Для отправки пакетов ваше устройство использует полученный немаршрутизируемый адрес («192.168.0.5»), но по мере прохождения пакетов через шлюз этот адрес заменяется фактическим маршрутизируемым адресом. Когда пакеты возвращаются на вашу рабочую станцию, маршрутизатор производит обратную замену.

Такой подход позволяет сохранять реальные маршрутизируемые адреса и многократно использовать одни и те же немаршрутизируемые адреса для рабочих станций, которые перемещаются из одной сети в другую.

4.7. Назначение глобального IP-адреса

Для того чтобы подключить новую сеть к интернету, необходимо обратиться к интернет-провайдеру, который осуществит соединение. Он предоставит вам диапазон IP-адресов (один или несколько сетевых номеров), которые можно будет назначать устройствам, подклю-

ченным к вашей сети. Интернет-провайдер предоставляет вам часть сетевых номеров, полученных от интернет-провайдера более высокого уровня.

На высшем уровне распределения IP-адресов находятся пять *региональных интернет-реестров* (Regional Internet Registries – RIR). Каждый из пяти реестров выделяет IP-адреса для одной географической области. Каждой точке Земли может быть присвоен сетевой номер одного из пяти реестров. Описанные реестры включают: Северную Америку (ARIN), Южную и Центральную Америку (LACNIC), Европу (RIPE NCC), Азиатско-Тихоокеанский регион (APNIC) и Африку (AFRNIC).

Когда были изобретены классические адреса IPv4, такие как «212.78.1.25», к интернету было подключено только несколько тысяч компьютеров. Тогда человечество и представить не могло, что когда-нибудь к интернету сможет подключиться миллиард устройств. На сегодняшний день, по мере развития интернета и интернета вещей, умным автомобилям, холодильникам, термостатам и даже лампам требуются собственные IP-адреса. Так, нам необходимо подключить к интернету более миллиарда устройств. Для того чтобы это стало возможным, инженеры разработали интернет-протокол нового поколения под названием «IPv6». 128-битные адреса IPv6 намного длиннее 32-битных адресов IPv4.

Региональные интернет-реестры (RIR) постепенно переходят от IPv4 к IPv6. Однако этот процесс, вероятнее всего, потребует много времени, в течение которого IPv4 и IPv6 должны нормально сосуществовать.

4.8. Заключение

IP-уровень позволяет нам осуществлять не единичные переходы (как на канальном уровне), а серии переходов, что позволяет быстро и эффективно маршрутизировать пакеты. IP-уровень позволяет оперативно реагировать на сбои сети и решать возникающие проблемы, таким образом поддерживая построение оптимальных путей маршрутизации для пакетов. К слову, для этого IP-уровню не требуется информационный центр маршрутизации. Каждый маршрутизатор исследует свое положение в общей сети и, взаимодействуя со своими соседями, способствует эффективной маршрутизации пакетов.

IP-уровень не дает нам 100%-ную гарантию доставки пакетов. Они могут быть потеряны из-за кратковременных разрывов подключения

или различного рода путаниц. Пакеты, которые ваша система отправляет позже, могут найти более быстрый маршрут и прибыть раньше пакетов, которые ваша система отправила раньше.

Если бы IP-уровень был создан таким образом, чтобы он никогда не терял пакеты, для него стало бы практически невозможным справиться со всеми сложностями, возникающими при работе с таким количеством устройств.

Для того чтобы решить проблемы IP-уровня, существует другой уровень – транспортный.

4.9. Глоссарий

Магистральный маршрутизатор – маршрутизатор, который пересылает трафик в ядре интернета.

DHCP (Dynamic Host Configuration Protocol) – протокол динамической настройки узла. Протокол, позволяющий портативным устройствам получать IP-адрес при смене местоположения.

Пограничный маршрутизатор – маршрутизатор, обеспечивающий соединение между локальной и глобальной сетями (то же, что и шлюз).

Идентификатор хоста – часть IP-адреса, которая используется для идентификации устройства в локальной сети.

IP-адрес – адрес, позволяющий устройству подключаться к глобальной сети и взаимодействовать с другими устройствами, имеющими IP-адреса. Для упрощения маршрутизации в ядре интернета IP-адреса разбиваются на сетевые номера и идентификаторы хоста. Примером IP-адреса может быть «212.78.1.25».

NAT (Network Address Translation) – преобразование сетевых адресов. Механизм, позволяющий использовать глобальный IP-адрес для множества устройств в одной локальной сети.

Номер сети – часть IP-адреса, используемая для определения локальной сети, к которой подключено устройство.

«Маршрутный вихрь» – сбой, при котором образуется бесконечный цикл передачи пакетов между некоторым количеством маршрутизаторов. Возникает при сбоях в таблицах маршрутизации.

RIR (Regional Internet Registries) – региональные интернет-реестры. Реестры, примерно соответствующие пяти существующим на Земле континентам, определяют IP-адреса для основных географических зон мира.

Таблицы маршрутизации – информация, хранящаяся в каждом маршрутизаторе и позволяющая отслеживать, какой исходящий канал следует использовать для каждого номера сети.

Время жизни данных (TTL – Time To Live) – значение, содержащееся в каждом пакете, по мере прохождения пакета через маршрутизаторы уменьшающееся на единицу. Когда TTL достигает нуля, пакет отбрасывается.

Traceroute – команда, доступная во многих системах Linux/UNIX, функция которой состоит в приблизительном определении маршрута пакета. В системах Windows может носить название tracert.

«Сеть с двумя соединениями» – сеть, которой характерно наличие как минимум двух возможных маршрутов между любой парой узлов в сети. При разрыве одного из соединений в такой сети общее соединение не будет затронуто.

4.10. Контрольные вопросы

Вы можете пройти этот тест онлайн по адресу **http://www.net-intro.com/quiz/**.

1. Какова основная функция сетевого уровня?
   А. Перемещение пакетов от исходного устройства к конечному путем нескольких переходов.
   Б. Перемещение пакетов через одно физическое соединение.
   В. Обработка отказа веб-сервера.
   Г. Шифрование конфиденциальных данных.

2. Сколько физических каналов проходит пакет от исходного до конечного устройства?
   А. 1.
   Б. 4.
   В. 15.
   Г. 255.

3. Какой из перечисленных ниже вариантов ответов является IP-адресом?
   А. 0f:2a:b3:1f:b3:1a.
   Б. 192.168.3.14.
   В. www.khanacademy.com.
   Г. @drchuck.

4. Зачем нужен переход от IPv4 на IPv6?
   А. В IPv6 меньше таблицы маршрутизации.

Б. IPv6 сокращает количество необходимых переходов пакета.
В. У человечества заканчиваются адреса IPv4.
Г. Адреса IPv6 – выбор производителей сетевого оборудования

5. Что такое номер сети?
   А. Группа IP-адресов с одинаковым префиксом.
   Б. GPS-координаты конкретной локальной сети.
   В. Количество переходов, которое требуется для прохождения пакета через сеть.
   Г. Общая задержка пакетов, проходящих через сеть.

6. Сколько устройств может иметь адрес в сети с номером «218.78.»?
   А. 650.
   Б. 6500.
   В. 65 000.
   Г. 650 000.

7. Как маршрутизаторы определяют путь пакета через интернет?
   А. Маршруты контролируются IRG (Группа интернет-маршрутизации).
   Б. Каждый маршрутизатор просматривает пакет и пересылает его, основываясь на своем лучшем предположении относительно нужного исходящего канала.
   В. Каждый маршрутизатор отправляет все пакеты по каждому исходящему каналу (алгоритм лавинной рассылки).
   Г. Каждый маршрутизатор удерживает пакет до тех пор, пока пакет не поступит от конечного устройства.

8. Что такое таблица маршрутизации?
   А. Список IP-адресов, сопоставленных с адресами каналов.
   Б. Список IP-адресов, сопоставленных с координатами GPS.
   В. Список сетевых номеров, сопоставленных с координатами GPS.
   Г. Список сетевых номеров, сопоставленных с исходящими каналами от маршрутизатора.

9. Как недавно подключенный к сети маршрутизатор заполняет свои таблицы маршрутизации?
   А. Консультируется с IANA (Управление присвоения номеров интернета).
   Б. Скачивает документы маршрутизации RFC (описание протоколов).

В. Обращается в Инженерную группу интернета (IETF).
Г. Обращается к соседним маршрутизаторам.

10. Что делает маршрутизатор, когда физический канал выходит из строя?
    А. Удаляет все записи таблицы маршрутизации для данного канала.
    Б. Консультируется со службой карт интернета (IMAP).
    В. Доменное имя (DNS) ищет IP-адрес.
    Г. Отправляет все пакеты для данного канала на исходное устройство.

11. В чем преимущество «сетей с двумя соединениями»?
    А. Таблицы маршрутизации намного меньше.
    Б. Они устраняют необходимость в сетевых номерах.
    В. Они поддерживают больше IPv4-адресов.
    Г. Сеть продолжает работать, даже когда один из каналов выходит из строя.

12. Все ли пакеты одного сообщения проходят через интернет по одному и тому же маршруту?
    А. Да.
    Б. Нет.

13. Как маршрутизаторы обнаруживают новые маршруты и улучшают свои таблицы маршрутизации?
    А. Каждый день в полночь загружают новую карту интернета по протоколу IMAP.
    Б. Периодически запрашивают сетевые таблицы у соседних маршрутизаторов.
    В. Случайным образом отбрасывают пакеты для запуска кода исправления ошибок в интернете.
    Г. Конечные устройства получают данные о скорости передачи.

14. Для чего пакету необходимо значение «время жизни»?
    А. Чтобы убедиться, что пакеты не попадут в маршрутную петлю.
    Б. Чтобы отслеживать, сколько минут требуется пакету для прохождения через сеть.
    В. Для поддержания соответствия между номерами сетей и координатами GPS.
    Г. Чтобы сообщить маршрутизатору подходящий выходной канал для определенного пакета.

15. Как работает утилита traceroute?
    А. Отправляет серию пакетов с низкими значениями TTL, чтобы получить информацию о том, где отбрасываются пакеты.
    Б. Загружает сетевой маршрут из карты интернета (IMAP).
    В. Связывается с сервером доменных имен, чтобы получить маршрут для определенного сетевого номера.
    Г. Просит маршрутизаторы добавить информацию о маршруте к пакету, поскольку он направляется от источника к месту назначения.

16. Сколько времени требуется, чтобы пакет пересек Тихий океан по подводному оптоволоконному кабелю?
    А. 0,0025 с.
    Б. 0,025 с.
    В. 0,250 с.
    Г. 2,5 с.

17. Как подключенное к сети Wi-Fi устройство получает IP-адрес?
    А. Использует DHCP.
    Б. Использует DNS.
    В. Использует HTTP.
    Г. Использует IMAP.

18. Какова функция Network Address Translation (NAT)?
    А. Ищет IP-адрес, связанный с текстовыми именами, такими как «www.dr-chuck.com».
    Б. Позволяет трафику IPv6 проходить через сети IPv4.
    В. Ищет лучший исходящий канал для определенного маршрутизатора и номера сети.
    Г. Повторно использует специальные сетевые номера, такие как «192.168.», на нескольких сетевых шлюзах в разных местах.

19. Как глобально управляются IP-адреса и сетевые номера?
    А. Существует пять реестров высшего уровня, которые управляют номерами сетей в пяти географических регионах.
    Б. IP-адреса назначаются во всем мире случайным образом по принципу лотереи.
    В. IP-адреса назначаются производителями сетевого оборудования.
    Г. IP-адреса основаны на координатах GPS.

20. Насколько IPv6-адреса длиннее, чем IPv4-адреса?
    А. Одинакового размера.
    Б. Адреса IPv6 на 50 % длиннее, чем адреса IPv4.
    В. Адреса IPv6 вдвое длиннее, чем адреса IPv4.
    Г. Адреса IPv6 в 10 раз длиннее, чем адреса IPv4.

21. О чем говорит IP-адрес, начинающийся с «169.»?
    А. Ваше подключение к интернету поддерживает протокол многоадресной рассылки.
    Б. Шлюз сопоставляет ваш локальный адрес с глобальным адресом с помощью NAT.
    В. Не было доступного шлюза для пересылки ваших пакетов в интернет.
    Г. Шлюз для данной сети – это низкоскоростной шлюз с небольшим размером окна.

22. Если вы начинаете работу с интернет-провайдером в Польше, какой блок IP-адресов назначит вам региональный интернет-реестр (RIR)?
    А. ARIN.
    Б. LACNIC.
    В. RIPE NCC.
    Г. APNIC.
    Д. AFRNIC.
    Е. United Nations.

Глава 5

Система доменных имен

Система доменных имен позволяет вам получать доступ к веб-сайтам по их доменному имени (например, **www.khanacademy.org**). Благодаря этому вам не нужно вести список числовых IP-адресов (например, «212.78.1.25»). IP-адрес определяется тем, *где* ваше устройство подключено к интернету. Когда вы пользуетесь портативным компьютером из разных мест, в каждом новом месте он получает новый IP-адрес. Поскольку к вашему портативному компьютеру никто не подключается, не имеет значения, меняется ли ваш IP-адрес время от времени. Но, например, к веб-серверу подключается очень много устройств, поэтому было бы неудобно, если бы он перемещался и менял IP-адрес.

Когда ваше устройство подключается к системе, используя адрес доменного имени, оно ищет IP-адрес, соответствующий данному доменному имени (далее DNS), а затем устанавливает соединение, используя нужный IP-адрес.

Такой подход упрощает процесс перемещения сервера из одного места в другое. Серверу присваивается новый IP-адрес, и данные адреса домена обновляются. После обновления записи DNS новым запросам на доменное имя дается новый IP-адрес. Поскольку конечные пользователи получают доступ к большинству серверов с использованием доменных имен и не видят IP-адрес, сервер можно переместить в новое сетевое соединение, что не повлияет на возможность доступа конечного пользователя к серверу.

5.1. Распределение доменных имен

В предыдущем разделе мы говорили о том, что IP-адреса назначаются в зависимости от того, где вы подключаете новую сеть к интернету. Распределение доменных имен определяется правилами организаций, которые «владеют» ими. На вершине ие-

рархии владения доменными именами находится организация под названием *Корпорация по управлению доменными именами и IP-адресами* (Internet Corporation for Assigned Network Names and Numbers – ICANN). ICANN распределяет *домены верхнего уровня* (top-level domain – TLD), такие как .com, .edu и .org между другими организациям. Недавно стал доступен новый набор TLD – .club и .help.

ICANN также занимается назначением двухбуквенных доменных имен верхнего уровня, таких как .us, .za, .nl и .jp, странам по всему миру. Такие доменные имена принято называть *Национальными доменами верхнего уровня* (country code top-level domain – ccTLD). Также часто используются TLD второго уровня, например .co.uk для коммерческих организаций в Великобритании. Политика подачи заявок на доменные имена с любым конкретным ccTLD в разных странах сильно различается.

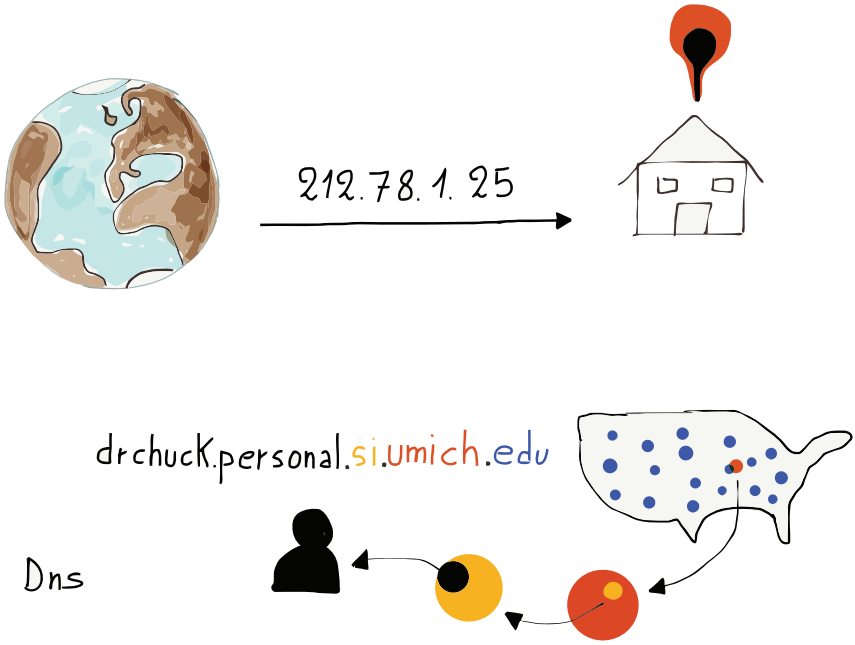

Рис. 5.1. Доменные имена

После назначения организации определенного доменного имени контролирующая организация может назначать поддомены внут-

ри домена верхнего уровня. Например, домен верхнего уровня .edu назначен организации Educause. Educause присваивает высшим учебным заведениям такие домены, как umich.edu. Как только Мичиганский университет получает контроль над umich.edu, он может самостоятельно выбирать поддомены в своем новом домене. Физические лица могут приобретать доменные имена с .com и .org. Владельцам таких доменов разрешается управлять своим доменом и создавать в нем поддомены для собственного или общего пользования.

5.2. Чтение доменных имен

В IP-адресе типа «212.78.1.25» префикс слева представляет собой «номер сети», поэтому в некотором смысле мы читаем IP-адреса слева направо, где левая часть IP-адреса несет в себе общую информацию, а правая часть адреса – уточняющую:

```
212.78.1.25
слева ----> направо.
```

Доменные имена мы читаем справа налево:

```
drchuck.personal.si.umich.edu
налево <---- справа.
```

Данный пример содержит «.edu», что означает, что организация, владеющая доменом, является высшим учебным заведением. Поддомен «umich.edu» отсылает нас к конкретному вузу.

5.3. Заключение

Хотя доменные имена и не являются одним из четырех уровней модели, рассматриваемой в данной книге, они значительно упрощают работу в интернете. DNS позволяют конечным пользователям использовать текстовые имена для серверов вместо числовых IP-адресов. Благодаря службе, которая сопоставляет доменные имена с IP-адресами, мы можем перемещать серверы с одного интернет-соединения на другое, не требуя от пользователей вручную изменять конфигурации для подключения к серверу.

Если вы хотите приобрести доменное имя для себя или своей компании, можете выбрать любое количество регистраторов доменных имен.

5.4. Глоссарий

DNS (Domain Name System) – система доменных имен. Система протоколов и серверов, которые позволяют сетевым приложениям искать доменные имена и получать соответствующий для него IP-адрес.

Доменное имя – имя, присвоенное в домене верхнего уровня. Например, khanacademy.org – это домен, который назначается в домене верхнего уровня «.org».

ICANN (Internet Corporation for Assigned Names and Numbers) – Корпорация по управлению доменными именами и IP-адресами. Назначает домены верхнего уровня для интернета и управляет ими.

Регистратор – компания, которая может регистрировать, продавать и размещать доменные имена.

Поддомен – имя, созданное «под» доменным именем. Например, «umich.edu» – это доменное имя, а «www.umich.edu» и «mail.umich.edu» являются поддоменами в «umich.edu».

TLD (top-level domain) – домен верхнего уровня. Правая часть доменного имени. Примеры TLD: «.com», «.org» и «.ru». Недавно были добавлены новые домены верхнего уровня, такие как «.club» и «.help».

5.5. Контрольные вопросы

Вы можете пройти этот тест онлайн по адресу **http://www.net-intro.com/quiz/**.

1. Каково назначение системы доменных имен?
   - А. Она позволяет подключенным к сети устройствам использовать текстовое имя и искать их IP-адреса.
   - Б. Она отслеживает GPS-координаты всех серверов.
   - В. Позволяет региональным интернет-реестрам (RIR) управлять IP-адресами на разных континентах.
   - Г. Она назначает разные IP-адреса портативным компьютерам при переходе от одного Wi-Fi к другому.

2. Какая организация назначает домены верхнего уровня, такие как «.com», «.org» и «.club»?
   - А. IANA (Управление присвоения номеров интернета).
   - Б. IETF (Инженерная группа интернета).

В. ICANN (Корпорация по управлению доменными именами и IP-адресами).
Г. IMAP (Internet Message Access Protocol).

3. Какой из вариантов является доменным адресом?

А. 0f:2a:b3:1f:b3:1a.
Б. 192.168.3.14.
В. www.khanacademy.org.
Г. @drchuck.

4. Что из этого **не** может делать владелец домена?

А. Создавать поддомен.
Б. Продавать поддомены.
В. Создавать новые домены верхнего уровня.
Г. Назначать IP-адрес домену или поддомену.

Глава 6

Транспортный уровень

Следующим уровнем интернет-архитектуры после сетевого является транспортный. Главная особенность этого уровня состоит в том, что он не гарантирует доставку каждого отдельного пакета. В большинстве случаев сетевой уровень работает практически идеально, но иногда возникают проблемы, например потери или неправильная маршрутизация пакетов.

Рис. 6.1. Транспортный уровень

Однако пользователям необходимо отправлять через интернет целые файлы или сообщения с гарантией того, что часть данных не потеряется в пути. На сегодняшний день пользователям требуется передача и отправка не только отдельных пакетов данных. Для того чтобы восстановить сообщение на конечном устройстве, все пакеты необходимо собрать в правильном порядке. Именно поэтому сеть должна уметь работать с ситуациями, когда пакеты приходят не по порядку или же вовсе теряются. Транспортный уровень обеспечивает восстановление исходных сообщений на конечном устройстве.

Как и IP-уровень, транспортный уровень решает проблемы повторной передачи пакетов посредством добавления к каждому пакету небольшого объема данных.

6.1. Заголовки пакетов

Как правило, пакеты состоят из заголовка канала, IP-заголовка, заголовка протокола управления передачи данных (TCP), а также определенного набора данных.

Заголовок канала	IP-заголовок	TCP-заголовок	Пакет данных
От/Кому	От/Кому/TTL	Порт/Относительный адрес	.....

Рис.6.2. Заголовки и данные

По завершении перехода пакета по определенному каналу заголовок старого канала отбрасывается, и на смену ему приходит заголовок нового, по которому пакету предстоит перейти. IP- и TCP-заголовки остаются неизменными. Помните, что по мере транспортировки пакет может проходить через несколько типов канальных уровней.

IP-заголовок содержит адреса исходного и конечного устройств, а также время жизни (TTL) пакета. IP-заголовок устанавливается на исходном устройстве и не изменяется (кроме TTL), по мере того как пакет перемещается через различные маршрутизаторы.

TCP-заголовки указывают порядок данных в каждом пакете. Когда исходное устройство разбивает сообщение или файл на пакеты, оно отслеживает положение каждого пакета относительно начала сообщения или файла, а затем добавляет в каждый созданный и отправленный пакет его относительный адрес.

6.2. Повторная сборка и повторная передача пакетов

Для того чтобы правильно восстановить исходное сообщение, конечное устройство руководствуется относительными адресами пакетов. Благодаря такому подходу технологии транспортного уровня легко могут обрабатывать пакеты, приходящие не по порядку. Такие пакеты помещаются в буфер обмена, а устройство отмечает, что в восстанавливаемом сообщении наблюдается пробел. Затем, по прибытии, нужный пакет идеально вписывается в повторно собранное сообщение.

Перед ожиданием подтверждения получения пакетов конечным устройством исходное устройство может отправлять строго ограниченный объем данных, что помогает избежать перегрузки сети. Количество данных определяется значением размера окна.

Исходное устройство отслеживает получение подтверждений приема пакетов от конечного. Если это происходит быстро, оно увеличивает значение размера окна, если нет – уменьшает. Посредством регулировки размера окна передающее устройство может отправлять большие объемы данных по быстрым и не слишком загруженным соединениям. Также устройства могут регулировать нагрузку на сеть при отправке данных по медленным или сильно загруженным соединениям.

В случае потери пакета он никогда не будет доставлен на конечное устройство, и оно в свою очередь не отправит подтверждение о получении этих данных. В таких случаях исходное устройство отслеживает количество неподтвержденных данных и, когда это значение достигает определенной точки, прекращает отправку новых пакетов.

В такие моменты оба устройства находятся в ожидании. Исходное ждет подтверждения потерянного пакета (которое никогда не придет), а конечное ожидает самого потерянного пакета (который также не придет). Для того чтобы ожидание не длилось вечность, конечное устройство отслеживает время, прошедшее с момента получения последнего пакета данных. Если времени прошло слишком много, конечное устройство отправляет на исходное сообщение, в котором указывает, где именно последний раз были получены данные. Когда исходное устройство получает это сообщение, оно выполняет резервное копирование и повторно отправляет данные, начиная с того места, где было осуществлено последнее успешное получение данных конечным устройством.

Таким образом, конечное устройство, подтверждающее получение данных, не позволяет исходному устройству уйти далеко вперед (оно ограничивается размером окна). Когда обнаруживается потеря данных, конечное устройство отправляет исходному запрос на «резервное копирование и перезапуск». Данный алгоритм представляет собой относительно простой способ, позволяющий отправлять по сети большие сообщения или файлы без угрозы потери данных.

В процессе отправки данных технологии транспортного уровня отслеживают, насколько быстро устройство получает подтверждения и динамически регулирует значение размера окна. Таким образом,

появляется гарантия быстрой отправки данных при высокоскоростном соединении и медленной отправке при медленном/загруженном соединении.

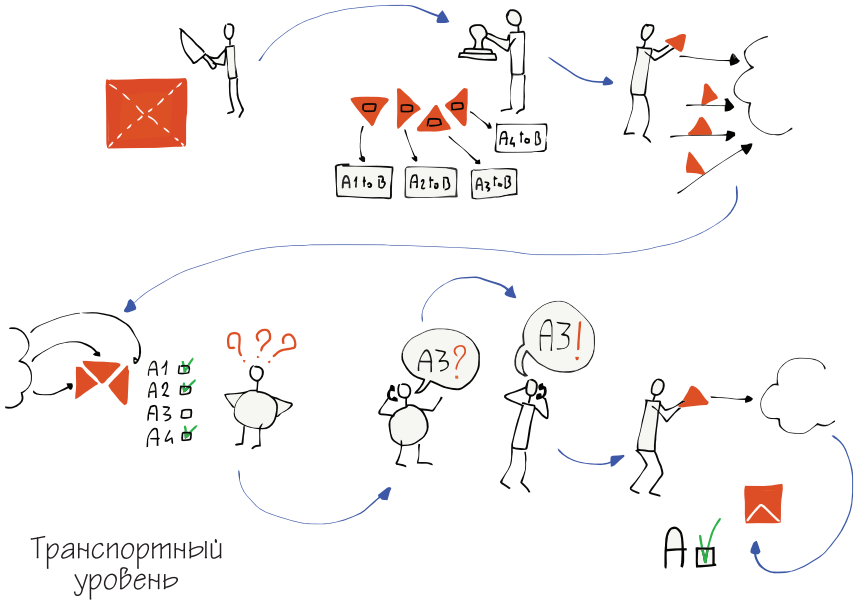

Рис. 6.3. Ожидание потерянного пакета

6.3. Транспортный уровень в действии

Среди особенностей транспортного уровня можно выделить то, что здесь исходное устройство должно хранить все данные для отправки до тех пор, пока они не будут подтверждены. Оно может отбросить отправленные данные только тогда, когда конечное устройство подтвердит их. На рис. 6.4 вы можете видеть сообщение, разбитое на множество пакетов. Первые десять пакетов данного сообщения были отправлены и затем подтверждены конечным устройством ('a'). Исходное устройство начало отправку еще шести пакетов ('S'), а затем остановило отправку, поскольку объем данных достиг установленного значения размера окна.

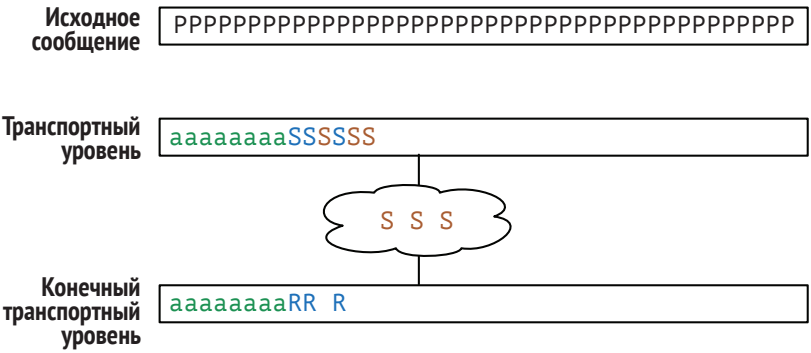

Рис. 6.4. Буферизация на транспортном уровне

Также на рисунке представлено еще три пакета, которые были отправлены, но еще не получены («S»). Ввиду большого количества переходов часто случается, что несколько пакетов приходит одновременно.

Конечное устройство получило и подтвердило десять пакетов, затем технологии транспортного уровня доставили их принимающему приложению ('a')[1]. Конечное устройство получило еще три пакета ('R'), но один пакет был потерян. Получение пакетов не по порядку не является серьезной проблемой, если недостающий пакет прибывает за достаточно короткий промежуток времени. Как только конечное устройство получает все пакеты, технологии транспортного уровня восстанавливают сообщение, соединяя пакеты вместе, как кусочки пазла, и только потом передают их принимающему приложению.

6.4. Клиентские и серверные приложения

Цель транспортного уровня заключается в обеспечении надежных соединений между сетевыми приложениями, чтобы они в свою очередь могли отправлять и получать потоки данных. Для приложения это так же просто, как запросить транспортный уровень установить соединение с приложением, запущенным на удаленном хосте. Мы называем приложение, которое инициирует соединение на локальном компьютере, «клиентом», а приложение, которое отвечает на запрос соединения, – «сервером». Комбинацию двух сетевых приложений, расположенных на разных концах соединения, называют приложением «клиент/сервер», так как эти две части неотделимы друг от друга.

[1] О технологиях прикладного уровня мы поговорим несколько позже.

Для того чтобы упростить установку соединения с удаленным устройством, а затем отправку и получение данных по этому соединению, на трех нижних уровнях интернет-архитектуры инженерами была проделана колоссальная работа.

6.5. Серверные приложения и порты

При установке соединения клиентского приложения с удаленным устройством важно, чтобы соединение выполнялось с правильным серверным приложением. На удаленном устройстве может одновременно работать любое количество различных серверных приложений, таких как:

- веб-серверы;
- видеосерверы;
- почтовые серверы.

Например, веб-клиент (браузер) должен подключиться к веб-серверу, запущенному на удаленном устройстве. В этом случае клиентскому приложению необходимо не только узнать, к какому серверу подключаться, но и выбрать конкретное приложение для взаимодействия.

Для того чтобы позволить клиентскому приложению выбирать серверное приложение, обычно используются порты. Они подобны добавочным номерам телефонов. Так, у всех добавочных номеров (серверных приложений) один и тот же основной номер (IP-адрес), а также у каждого из них также есть свой внутренний номер (номер порта).

При запуске серверного приложения входящие соединения на указанном порту «прослушиваются». После того как серверное приложение зарегистрирует свою готовность к приему входящих соединений, оно будет ждать установки первого соединения.

Существует список стандартных портов для различных серверных приложений, позволяющих клиентским приложениям найти нужный порт:

- Telnet (23) – Подключение;
- SSH (22) – Защищенное подключение;
- HTTP (80) – Всемирная паутина;
- HTTPS (443) – Защищенное подключение к Всемирной паутине;
- SMTP (25) – Входящая почта;
- IMAP (143/220/993) – Получение почты;
- POP (109/110) – Получение почты;

- DNS (53) – Распределение доменных имен;
- FTP (21) – Передача файлов.

Иногда серверы используют и нестандартные порты. Если вы занимаетесь веб-разработкой, можете запустить веб-сервер с нестандартным портом (3000, 8080 или 8888). Если вы видите URL-адрес, например

`http://testing.example.com:8080/login`,

то «8080» указывает, что ваш браузер будет использовать веб-протоколы для взаимодействия с сервером, но подключаться к порту 8080 вместо порта 80 по умолчанию.

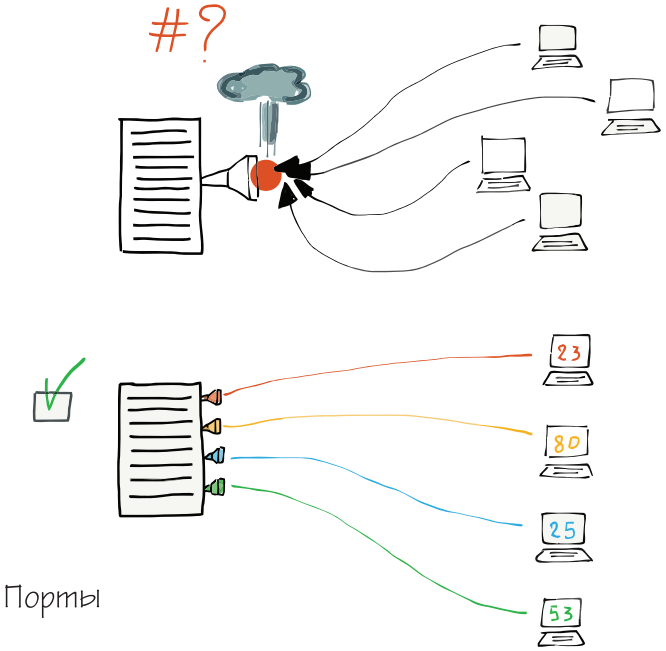

Рис. 6.5. TCP-порты

6.6. Заключение

В некотором смысле цель транспортного уровня заключается в компенсации возможной потери данных на канальном и сетевом уровнях. Когда эти уровни теряют или перенаправляют пакеты, транс-

портный уровень повторно собирает и/или повторно передает эти данные. Благодаря транспортному уровню два нижних уровнях могут игнорировать проблемы повторной передачи и ограничения скорости.

Отчасти цель многоуровневой архитектуры состоит также в том, чтобы разбить слишком сложную задачу на более мелкие подзадачи. Каждый уровень фокусируется на решении части общей проблемы и полагается на другие уровни, которые решают подзадачи, отведенные для них.

6.7. Глоссарий

Подтверждение получения пакета – отправка конечным устройством на исходное устройство сообщения о том, что данные были получены обратно.

Буферизация – временное хранение отправленных или полученных данных (до тех пор, пока устройство не убедится, что данные больше не понадобятся).

Прослушивание – процесс, начинающийся, когда серверное приложение запущено и готово принимать входящие соединения от клиентских приложений.

Порт – технология, позволяющая множеству различных серверных приложений ожидать входящих подключений на одном устройстве. Каждое приложение прослушивает отдельный порт. Клиентские приложения устанавливают соединения с хорошо известными номерами портов, чтобы убедиться, что они обращаются к правильному серверному приложению.

6.8. Контрольные вопросы

Вы можете пройти этот тест онлайн по адресу **http://www.net-intro.com/quiz/**.

1. Какова основная задача транспортного уровня (TCP)?
   А. Перемещать пакеты от исходного устройства к конечному посредством нескольких переходов.
   Б. Перемешать пакеты по одиночному физическому соединению.
   В. Работать с потерянными и поврежденными пакетами.
   Г. Разбираться с шифрованием конфиденциальных данных.
2. Из чего состоит заголовок TCP?
   А. Физический адрес.

Б. IP-адрес и время жизни.
В. Номер порта и относительный адрес.
Г. Информация о том, какой документ был запрошен.

3. Почему размер окна важен для правильного функционирования сети?
   А. Потому что слишком большие пакеты будут перегружать оптоволоконные соединения.
   Б. Потому что он не позволяет быстрому устройству отправлять слишком много данных при медленном соединении.
   В. Он ограничивает количество переходов, которые может выполнить пакет, прежде чем он будет отброшен.
   Г. Он определяет, какая часть IP-адреса является номером сети.

4. Что происходит, когда исходное устройство получает подтверждение от конечного?
   А. Исходное устройство повторно отправляет данные, чтобы убедиться, что они были переданы правильно.
   Б. Исходное устройство отправляет больше данных до размера окна.
   В. Исходное устройство отправляет «подтверждение для подтверждения».
   Г. Исходное устройство отправляет подтверждение в карту интернета (IMAP).

5. Какое из перечисленных устройств обнаруживает и принимает меры в случае потери пакетов?
   А. Исходное устройство.
   Б. Сетевой шлюз.
   В. Базовый интернет-маршрутизатор.
   Г. Конечное устройство.

6. Какое из перечисленных устройств сохраняет пакеты данных, чтобы их можно было повторно передать в случае потери?
   А. Исходное устройство.
   Б. Сетевой шлюз.
   В. Базовый интернет-маршрутизатор.
   Г. Конечное устройство.

7. Что из перечисленного больше всего похоже на TCP-порт?
   А. Железнодорожная станция.
   Б. Подводный сетевой кабель.

В. Номер квартиры.
Г. Сад скульптур.

8. Какая часть приложения «клиент/сервер» должна запускаться первой?
   А. Клиент.
   Б. Сервер.

9. Какой номер порта системы доменных имен (DNS)?
   А. 22.
   Б. 80.
   В. 53.
   Г. 143.

10. Какой номер порта протокола получения почты IMAP?
    А. 22.
    Б. 80.
    В. 53.
    Г. 143.

Глава 7

Прикладной уровень

Итак, мы достигли вершины сетевой модели TCP/IP. На прикладном уровне работает такое сетевое программное обеспечение, как веб-браузеры, программы электронной почты, видеоплееры и сетевые видеоплееры. Со всеми этими приложениями взаимодействуем мы (пользователи), а сами приложения взаимодействуют от нашего имени с сетью.

Рис.7.1. Прикладной уровень

7.1. Клиентские и серверные приложения

Важно помнить, что для работы сетевого приложения всегда требуются две части. Именно поэтому архитектура сетевого приложения называется «клиент/сервер». Серверная часть приложения работает где-то далеко в сети. Она содержит информацию, которую пользователи хотят просматривать или с которой взаимодействуют. Клиентская часть приложения устанавливает соединения с серверным приложением, извлекает информацию и предоставляет ее пользователю.

Для обмена данными обе стороны используют транспортный уровень (как на исходном, так и на конечном устройствах).

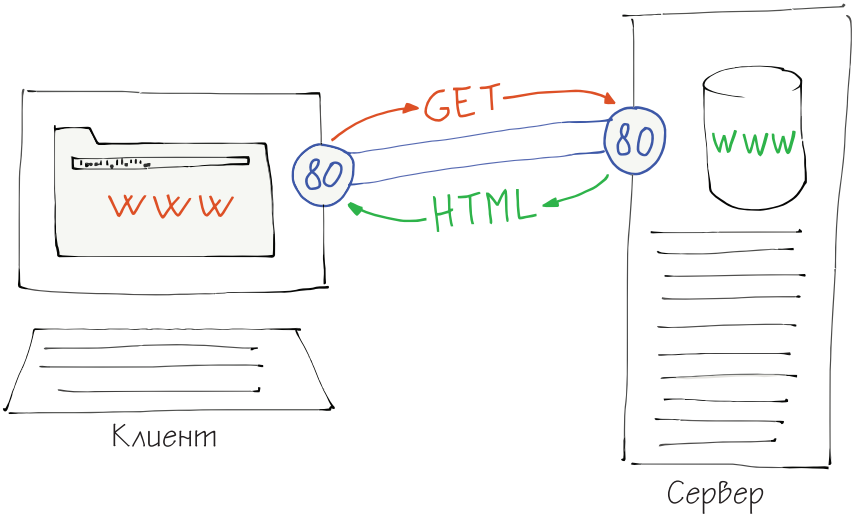

Рис.7.2. Клиент-серверные приложения

Например, для того чтобы перейти по адресу **www.khanacademy.org**, на вашем устройстве должно быть запущено веб-приложение. После того как вы введете адрес в адресную строку браузера, приложение на вашем устройстве подключится к соответствующему веб-серверу, извлечет страницы, которые вы можете просмотреть, а затем представит их вам.

Веб-браузер на вашем устройстве отправляет запрос на подключение к **www.khanacademy.org**. Затем устройство ищет доменное имя, чтобы найти соответствующий IP-адрес для сервера. После этого оно устанавливает транспортное соединение с этим IP-адресом и начинает запрашивать данные. По мере получения данных браузер отображает их пользователю. Иногда в окне браузера вы можете наблюдать небольшой анимированный значок, который означает, что данные загружаются.

На другом конце соединения находится другое приложение, которое называется «веб-сервер». Программа этого приложения всегда активна и ожидает входящих соединений. Так, когда вы хотите загрузить какую-либо веб-страницу, вы подключаетесь к уже запущенному серверному приложению и ожидаете вашего подключения.

В некотором смысле транспортный, сетевой и канальный уровни, наряду с системой доменных имен, работают как телефонная сеть для сетевых приложений. Они подключаются к различным серверным приложениям в сети и «общаются» с этими приложениями для обмена данными.

7.2. Протоколы прикладного уровня

Подобно людям, разговаривающим по телефону, каждая пара сетевых приложений руководствуется набором правил, регулирующих их разговор. В большинстве культур принято говорить «Привет», когда вы поднимаете телефонную трубку. Обычно звонящий (клиент) молчит, до тех пор пока его собеседник (сервер) не скажет «Привет». Если когда-нибудь у вас случалось так, что тот, кому вы звонили молчал, вы знаете, как это может сбить с толку. В таком случае вы, вероятно, думали, что возникли проблемы со связью, клали трубку и перезванивали.

Набор правил, регулирующих коммуникацию, называется протоколом. Слово «протокол» означает правило, в соответствии с которым регулируется порядок осуществления различной деятельности, особенно в области дипломатии. Основная мысль здесь состоит в том, что при формальном общении мы должны вести себя согласно четкому набору правил. Слово «протокол» применяется и к правилам, установленным для сетевых приложений. Машины любят точность, и без четкого свода правил они не смогут общаться.

Существует множество различных сетевых приложений, для каждого из которых необходим четко документированный протокол. Только так все серверы и клиенты смогут взаимодействовать друг с другом. Некоторые из таких протоколов имеют довольно сложную структуру.

Протокол, созданный для регулирования взаимодействия веб-браузера с веб-сервером, описан в ряде больших документов, начиная с **https://tools.ietf.org/html/rfc7230**.

Официальное название этого протокола – гипертекстовый транспортный протокол (HyperText Transport Protocol), или сокращенно – HTTP. Если URL-адрес, который вы вводите в браузере, начинается с «http:» или «https:», это значит, что вы хотите получить данные, используя протокол HTTP.

В разделе 5.3.2 на стр. 41 приведенного выше документа вы можете найти точное описание алгоритма отправки данных веб-клиентом на веб-сервер.

```
GET http://www.example.org/pub/WWW/TheProject.html HTTP/1.1
```

В сравнении с большинством протоколов приложения «клиент/сервер» HTTP достаточно прост. Однако, несмотря на простоту, в нем существует множество деталей, которые позволяют веб-клиентам и серверам эффективно взаимодействовать и передавать широкий спектр информации и данных. Сертификация HTTP состоит из шести отдельных документов и составляет в общей сложности 305 страниц. На первый взгляд, это очень много. Но не стоит забывать о том, что ключ к разработке эффективных протоколов заключается в рассмотрении всех возможных вариантов использования протокола и тщательном описании каждого сценария.

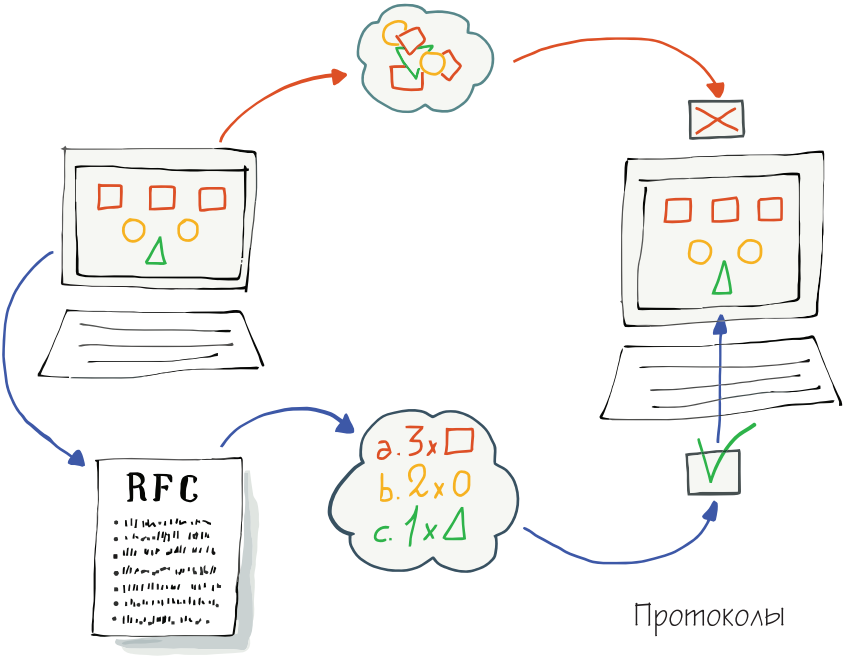

Рис.7.3. Протоколы прикладного уровня

7.3. Исследование протокола HTTP

В этом разделе мы притворимся веб-браузером и рассмотрим, как работает протокол HTTP с этой точки зрения. Для этого мы будем отправлять HTTP-команды на веб-сервер вручную. Чтобы поиграть

с протоколом HTTP, мы будем использовать одно из самых ранних интернет-приложений.

Приложение telnet было разработано в 1968 году в соответствии с одним из первых интернет-стандартов: **https://tools.ietf.org/html/rfc15**.

Его спецификация состоит всего из пяти страниц, и поэтому сейчас вы можете подумать, что с легкостью сможете прочитать ее и понять большую часть содержания. Telnet – настоящий динозавр среди клиентских приложений. В перспективе эпохи интернета оно появилось в «доисторические» времена. Интернет был создан в 1985 году в рамках проекта NSFNet, а предшественник NSFNet под названием ARPANET появился в 1969 году. Telnet был создан еще до того, как была запущена в производство первая сеть TCP/IP.

Однако стоит отметить, что telnet все еще присутствует в большинстве современных операционных систем. Так, вы можете запустить его из терминала (командной строки) Macintosh и Linux. Приложение telnet также присутствовало в Windows, начиная с Windows 95 и до Windows XP (в более поздних версиях его убрали). Если вы пользуетесь более поздней версией Windows, для того чтобы повторить примеры, приведенные в этом разделе, вы можете загрузить и установить telnet-клиент.

Telnet – это простое приложение. Запустите telnet из командной строки (или терминала) и введите следующую команду:

```
telnet www.dr-chuck.com 80
```

Первый параметр – это доменное имя (также здесь подойдет IP-адрес) и порт для подключения на этом хосте. Порт необходим для того, чтобы указать, к какому серверу приложений мы хотим подключиться. На порту 80 обычно можно найти HTTP-приложение (веб-сервера) на хосте. Если веб-сервер не прослушивает порт 80, соединение прервется, и возникнет уведомление об ошибке. Но, если все получится, мы подключимся к этому веб-серверу, и все, что мы введем на клавиатуре, будет отправлено непосредственно туда. На этом этапе крайне важно знать, как правильно вводить HTTP-команды. Если мы введем что-то неправильно, может возникнуть подобная ошибка:

```
telnet www.dr-chuck.com 80
Trying 198.251.66.43...
Connected to www.dr-chuck.com.
Escape character is '^]'.
HELP
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
```

```
<html><head>
<title>501 Method Not Implemented</title>
...
</body></html>
Connection closed by foreign host.
```

Набирая telnet в терминале, мы запрашиваем соединение с портом 80 (веб-сервер) на хосте **www.dr-chuck.com**. Здесь мы можем видеть, как наш транспортный уровень ищет имя домена, обнаруживает, что фактический адрес – «198.251.66.43», а затем устанавливает соединение с этим сервером. После установки соединения сервер ждет, пока мы введем команду, за которой последует нажатие **Ввода**. Поскольку протокол нам неизвестен, мы набираем «HELP» и нажимаем **Ввод**. Сервер в свою очередь выдает сообщение об ошибке и закрывает соединение. Второго шанса у нас нет. Если мы не знаем, как правильно выполнять запросы, сервер не захочет с нами общаться.

Но давайте все же вернемся к разделу 5.3.2 документа RFC-7230 и попробуем запросить документ снова, используя правильный синтаксис:

```
telnet www.dr-chuck.com 80
Trying 198.251.66.43...
Connected to www.dr-chuck.com.
Escape character is '^]'.
GET http://www.dr-chuck.com/page1.htm HTTP/1.0

HTTP/1.1 200 OK
Last-Modified: Sun, 19 Jan 2014 14:25:43 GMT
Content-Length: 131
Content-Type: text/html

<h1>The First Page</h1>
<p>
If you like, you can switch to the
<a href="http://www.dr-chuck.com/page2.htm">
Second Page</a>.
</p>
Connection closed by foreign host.
```

Так, мы снова подключаемся к веб-браузеру с помощью telnet, а затем отправляем команду GET, которая указывает, какой документ мы хотим получить. Здесь мы используем HTTP 1.0, поскольку эта

версия несколько проще, чем HTTP 1.1. Затем нажимаем **Ввод** и отправляем пустую строку, чтобы отметить окончание запроса.

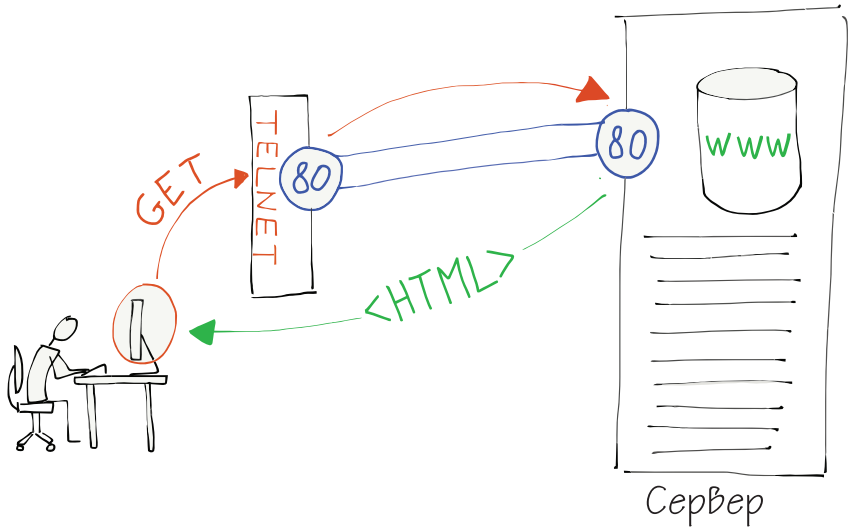

Рис.7.4. Отправка HTTP команд вручную

Теперь, когда мы отправили правильный запрос, хост отвечает серией заголовков, описывающих документ, за которыми следует пустая строка, и после чего отправляет сам документ.

В заголовках содержатся метаданные (данные о данных) запрашиваемого нами документа. Например, первая строка содержит код состояния.

В приведенном примере код состояния «200» означает, что все прошло успешно. Код «404» в первой строке заголовком указывает на то, что запрошенный документ не найден. «301» указывает на то, что документ перемещен в другое место.

Коды состояния для HTTP сгруппированы в следующие классы: коды 2xx указывают на то, что запрос был осуществлен успешно, коды 3xx указывают на перенаправление, коды 4xx указывают на ошибку клиента, а коды 5xx – на ошибку сервера.

В примере приведен документ на языке гипертекстовой разметки (HyperText Markup Language – HTML), поэтому он содержит такие теги, как <h1> и <p>. Когда ваш браузер получает документ в формате HTML, он просматривает разметку в документе, интерпретирует ее и представляет вам отформатированную версию документа.

7.4. Протокол доступа к электронной почте IMAP

Протокол HTTP – лишь один из многих существующих протоколов клиент-серверных приложений. Существует, например, другой распространенный протокол, который используется для того, чтобы почтовое приложение, работающее на вашем устройстве, могло получать почту с центрального сервера. Поскольку ваш персональный компьютер (или другое устройство) не всегда активен, вся ваша почта отправляется на сервер и хранится там до тех пор, пока вы не включите свой компьютер и не получите письмо.

Как и многие стандарты приложений, протокол доступа к сообщениям в интернете (Internet Message Access Protocol – IMAP) описан в серии документов *RFC* (Request For Comment), начиная с этого:

`https://tools.ietf.org/html/rfc3501`.

IMAP несколько сложнее, чем веб-протокол, поэтому мы уже не сможем использовать команду telnet. Но, если вы, например, собирались разработать приложение для чтения почты, вы могли бы внимательно прочитать этот документ и разработать код для успешного взаимодействия с совместимым со стандартами сервером IMAP. Вот простой пример из раздела 6.3.1 вышеупомянутого документа, показывающий, что отправляет клиент (C:) и как реагирует сервер (S:):

```
C: A142 SELECT INBOX
S: * 172 EXISTS
S: * 1 RECENT
S: * OK [UNSEEN 12] Message 12 is first unseen
S: * OK [UIDVALIDITY 3857529045] UIDs valid
S: * OK [UIDNEXT 4392] Predicted next UID
S: * FLAGS (\Answered \Flagged \Deleted \Seen \Draft)
S: * OK [PERMANENTFLAGS (\Deleted \Seen \*)] Limited
S: A142 OK [READ-WRITE] SELECT completed
```

Сообщения клиента и сервера не предназначены для просмотра конечным пользователем, поэтому для человека они не особо содержательны. Такие сообщения точно отформатированы и отправляются в определенном порядке, благодаря чему они могут быть сгенерированы и прочитаны сетевыми компьютерными приложениями на обоих концах соединения.

7.5. Управление потоками

При рассмотрении транспортного уровня мы говорили о значении размера окна, которое представляло собой объем данных, которые мог отправить исходный компьютер перед ожиданием подтверждения.

На рис. 7.5 представлено сообщение, разбитое на пакеты, часть которых была отправлена и подтверждена. Шесть отправленных пакетов, изображенных на рисунке, еще не были подтверждены, поскольку исходное устройство достигло предельного значения размера окна и приостановило отправку пакетов до получения подтверждения от конечного устройства. Конечное устройство получило три пакета, один из которых был поврежден.

При рассмотрении этого примера с точки зрения транспортного уровня мы не говорили о том, откуда приходили пакеты для отправки и куда они отправлялись по приходу на конечное устройство. Теперь, когда мы рассматриваем прикладной уровень, самое время поговорить о приложениях, которые как раз и являются источником и получателем потоков данных.

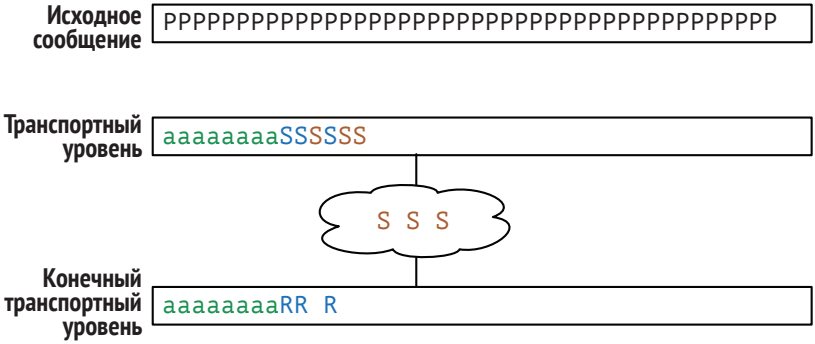

Рис.7.5. Буферизация на транспортном уровне

Предположим, веб-браузер установил транспортное соединение с веб-сервером и начал загрузку файла изображения. После этого веб-сервер открывает файл изображения и отправляет данные из файла на свой транспортный уровень. Однако технологии транспортного уровня руководствуются размером окна, поэтому сервер может отправить ограниченный объем данных за один раз. По достижении предельного значения размера окна веб-сервер приостанавливает отправку данных до тех пор, пока транспортный уровень на конечном устройстве не начнет принимать и подтверждать пакеты.

Когда транспортный уровень на конечном устройстве начинает принимать пакеты, восстанавливать поток данных и подтверждать получение данных, он доставляет уже восстановленный поток пакетов в приложение веб-браузера, отображаемое на экране пользователя. Иногда при медленном соединении вы можете видеть, как ваш браузер «рисует» картинки по мере загрузки данных. При быстром подключении данные поступают настолько быстро, что изображения появляются мгновенно.

Если поместить изображение пакетов на транспортном уровне в центр, а по краям добавить серверное и клиентское приложения, то мы получим следующую картину:

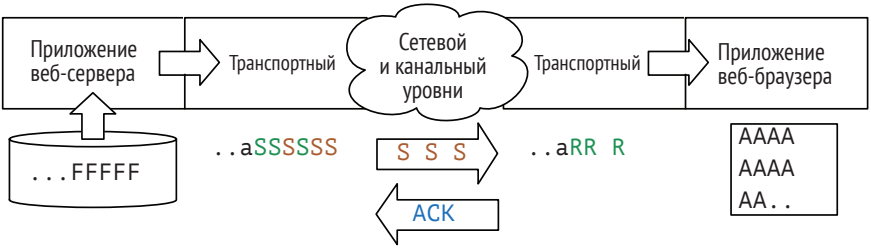

Рис.7.6. Буферизация на прикладном и транспортном уровнях

Веб-сервер считывает файл изображения ('F') и отправляет его в веб-браузер в виде потока данных с максимально возможной скоростью. Исходный транспортный уровень разбивает поток на пакеты и отправляет их на конечное устройство с помощью IP-адреса.

Транспортный уровень отправляет шесть пакетов ('S'), а затем прекращает отправку, поскольку достиг предельного значения размера окна, работа веб-сервера была приостановлена. Три из этих шести пакетов дошли до транспортного уровня на конечном устройстве ('R'), а три других пакетов все еще находятся в пути ('S').

Поскольку транспортный уровень конечного устройства снова разделяет поток на пакеты, он отправляет подтверждение (ACK) и доставляет данные принимающему приложению (веб-браузеру). Он в свою очередь воссоздает изображение ('A') и отображает его пользователю по мере получения данных.

Ключевой момент, на который следует обратить внимание на этом рисунке, состоит в том, что технологии транспортного уровня обоих устройств хранят далеко не все пакеты. Они сохраняют только те, которые находятся «в пути» и не подтверждены. После доставки и под-

твердения в конечное приложение и исходное, и конечное устройство могут отбросить пакеты.

Когда на исходное устройство приходит подтверждение от конечного, первое возобновляет работу приложения веб-сервера, и веб-сервер продолжает считывать данные из файла и отправлять их на исходный транспортный уровень для передачи.

Возможность запускать и останавливать исходное приложение называется управлением потоков. С помощью этой возможности мы может убедиться, что данные отправляются на максимально возможной скорости, но при этом не перегружают сеть. Приложения не занимаются управлением потока, они лишь стараются отправить или получить данные настолько быстро, насколько это возможно. Технологии транспортного уровня на исходном и конечном устройствах запускают и останавливают приложения по мере необходимости, ориентируясь на скорость и надежность сети.

7.6. Создание сетевых приложений

Сетевые приложения создаются на одном или нескольких языках программирования. Многие из таких языков имеют библиотеки кода, что значительно упрощает написание сетевых приложений. С помощью хорошей программной библиотеки установить соединение с приложением, работающим на сервере, отправить данные на сервер и получить данные с сервера, как правило, так же просто, как прочитать или написать файл.

В качестве примера на рисунке ниже приведен код, написанный на языке программирования Python. К слову, это все, что потребуется для подключения к веб-серверу и получения документов.

```
import socket

mysock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
mysock.connect(('www.py4inf.com', 80))
mysock.send('GET http://www.py4inf.com/code/romeo.txt HTTP/1.0\n\n')

while True:
    data = mysock.recv(512)
    if ( len(data) < 1 ) :
        break
    print data

mysock.close()
```

Рис.7.7. Программирование сокетов на Python

Знать язык программирования Python не обязательно. Однако факт остается фактом, что для создания и использования сетевого соединения вам потребуется всего десять строк кода приложения. Простота кода обусловливается тем, что транспортный, сетевой и канальный уровни способны справляться со своими задачами настолько эффективно, что приложения могут игнорировать почти все детали реализации сети.

При написании приложения на Python в строке кода

```
mysock.connect(('www.py4inf.com', 80))
```

мы указали, что приложение должно прослушивать входящие соединения на порту 80 на удаленном устройстве **www.py4inf.com**.

Выбирая порт 80, мы указываем, что на этом хосте мы хотим подключиться к серверу Всемирной сети, и ожидаем связи с этим сервером, используя HTTP.

7.7. Заключение

Три нижних уровня (транспортный, сетевой и канальный) нацелены на то, чтобы дать возможность приложениям работать только с задачами прикладного уровня. Таким образом, они берут на себя самые сложные части передачи данных по сети.

Такой подход упрощает создание сетевых приложений. Именно поэтому их количество растет. Среди широкого спектра сетевых приложений встречаются веб-браузеры, почтовые приложения, сетевые видеоигры, приложения сетевой телефонии и многие другие. Также стоит отметить, что все это дает возможности для экспериментов и позволяет создавать совершенно новые типы сетевых приложений.

7.8. Глоссарий

HTML (HyperText Markup Language) – язык гипертекстовой разметки. Язык текстового формата, в котором разметка текста происходит с помощью тегов с символами «меньше» и «больше». Например: `<p>This is <strong>nice</strong></p>`.

HTTP (HyperText Transport Protocol) – гипертекстовый транспортный протокол. Протокол прикладного уровня, который позволяет веб-браузерам получать веб-документы с веб-серверов.

IMAP (Internet Message Access Protocol) – протокол доступа к электронной почте, позволяющий почтовым клиентам входить в систему и получать почту с почтовых серверов с поддержкой IMAP.

Управление потоками – подход, при котором исходное устройство периодически замедляется с целью контроля перегрузки сети или конечного устройства. Также такой подход позволяет исходному устройству увеличивать скорость передачи данных в тех случаях, когда сеть и конечное устройство способны обрабатывать более высокие скорости передачи данных.

Сокет – программная библиотека, доступная на многих языках программирования, которая делает создание сетевого подключения и обмен данными почти таким же простым, как открытие и чтение файла на любом вашем устройстве.

Код состояния – один из аспектов HTTP, который призван указывать на выполнение или невыполнение запроса. Самый известный код состояния HTTP – «404». С его помощью HTTP-сервер сообщает HTTP-клиенту (т. е. браузеру), что запрошенный документ не может быть найден.

Telnet – простое клиентское приложение, которое устанавливает TCP-соединения с различными комбинациями адресов/портов и позволяет передавать введенные данные. На заре интернета telnet использовался для удаленного входа на устройство через сеть.

Веб-браузер – клиентское приложение, которое вы запускаете на своем устройстве для получения и отображения веб-страниц.

Веб-сервер – приложение, которое доставляет (обслуживает) веб-страницы.

7.9. Контрольные вопросы

Вы можете пройти этот тест онлайн по адресу **http://www.net-intro.com/quiz/**.

1. Какой уровень находится под прикладным?
   А. Транспортный.
   Б. Сетевой.
   В. Канальный.
   Г. Заторможенный.

2. Какой документ используется для описания широко используемых протоколов прикладного уровня?
   А. DHCP.
   Б. RFC.
   В. APPDOC.
   Г. ISO 9000.

3. Что из этого было создано на прикладном уровне?
   А. 0f:2a:b3:1f:b3:1a.
   Б. 192.168.3.14.
   В. www.khanacademy.com.
   Г. http://www.dr-chuck.com/.

4. Что из перечисленного не является задачей прикладного уровня?
   А. Решение вопроса о том, кто должен начинать говорить первым, – клиент или сервер.
   Б. Контроль формата команд и ответов, которыми обмениваются через сокет.
   В. Контроль за тем, как меняется значение размера окна при передаче данных через сокет.
   Г. Вопрос представления данных и обеспечение совместимости.

5. Что из перечисленного является протоколом прикладного уровня?
   А. HTTP.
   Б. TCP.
   В. DHCP.
   Г. Ethernet.

6. Какой порт обычно используется для установки соединения с веб-сервером?
   А. 23.
   Б. 80.
   В. 103.
   Г. 143.

7. Какую команду веб-браузер отправляет веб-серверу для получения веб-документа?
   А. RETR.
   Б. DOCUMENT.
   В. 404.
   Г. GET.

8. Какова роль заголовка «Content-type:» при получении документа по веб-протоколу?
   А. Он сообщает браузеру, как отображать полученный документ.
   Б. Он перенаправляет браузер на новый адрес документа.
   В. Он сообщает браузеру, есть ли содержимое в полученном пакете.
   Г. Он сообщает браузеру, где заканчиваются заголовки и начинается содержимое.

9. Какую стандартную команду UNIX можно использовать для отправки простых команд на веб-сервер?
   А. ftp.
   Б. ping.
   В. traceroute.
   Г. telnet.

10. Что означает код состояния HTTP «404»?
   А. Документ перемещен.
   Б. Документ найден.
   В. Ошибка протокола.
   Г. Документ не найден.

11. Какие символы используются для разметки HTML-документов?
   А. Знаки «меньше» и «больше» (<>).
   Б. Восклицательные знаки (!).
   В. Квадратные скобки ([]).
   Г. Фигурные скобки ({}).

12. Что из перечисленного является общим протоколом приложений для получения почты?
   А. RFC.
   Б. HTML.
   В. ICANN.
   Г. IMAP.

13. Какой протокол прикладного уровня описывает RFC15?
   А. telnet.
   Б. ping.
   В. traceroute.
   Г. www.

14. Что происходит с серверным приложением, которое отправляет большой файл, когда транспортный уровень достиг предельного значения размера окна, но еще не получил подтверждения?
   А. Приложение переключает передачу на новый сокет.
   Б. Приложение «вылетает», и его необходимо перезапустить.
   В. Приложение приостанавливается до тех пор, пока удаленное устройство не подтвердит получение данных.
   Г. Ближайший маршрутизатор-шлюз начинает отбрасывать пакеты, превышающие размер окна.

15. Что такое сокет?
    А. Способ подключения устройств к беспроводной сети.
    Б. Способ получения устройствами IP-адреса.
    В. Запись в таблице маршрутизации.
    Г. Двустороннее соединение для передачи данных между парой клиентских и серверных приложений.

16. Что необходимо приложению, чтобы установить соединение через сокет в программном обеспечении?
    А Адрес сервера и номер порта на сервере.
    Б. Маршрут между исходным и конечным устройствами.
    В. Знать, какая часть IP-адреса является номером сети.
    Г. Начальный размер окна TCP во время передачи.

Глава 8

Безопасность транспортного уровня

На заре интернета сети были небольшими, а все маршрутизаторы находились в безопасных местах. Каждое устройство, подключенное к интернету, защищало себя от установки нежелательных соединений, поэтому никто не заботился о защите передаваемых данных.

Канальный, сетевой и транспортный уровни были сосредоточены лишь на передаче данных и решении задач крупной распределенной сети.

К концу 1980-х годов популярность интернета возрастала, а в 1994 году случился настоящий интернет-бум. Именно в это время возникли вопросы безопасности и конфиденциальности сетевого трафика. С появлением возможности осуществлять покупки онлайн возникла необходимость пересылки по сети таких данных, как номера кредитных карт или банковских счетов. Так вопросы защиты данных вышли на первый план. Затем они стали еще острее, поскольку с развитием беспроводных технологий, таких как Wi-Fi, защита данных требовалась при выполнении любых действий в интернете.

На сегодняшний день существует два основных подхода к защите сетевой активности. Суть первого подхода состоит в том, что все сетевое оборудование (маршрутизаторы и каналы связи) физически находятся в безопасных местах, иными словами, никто не сможет контролировать трафик во время его прохождения по сети. Такой подход не представляется целесообразным при наличии большого количество маршрутизаторов, принадлежащих различным организациям. Несмотря на то что операторы маршрутизаторов придерживаются строгих алгоритмов выполнения процедур и следуют политике безопасности, рано или поздно проблемы все же возникнут. Например, как только вы решите подключиться к Wi-Fi, перехватить данные сможет любой злоумышленник.

Сущность второго подхода заключается в шифровании данных. Так, данные зашифровываются на исходном устройстве, передаются по физическому каналу и затем расшифровываются на конечном устройстве. При таком подходе злоумышленники, конечно, могут увидеть ваши данные, но расшифровать их – нет. Также шифрование гарантирует невозможность внести изменения в передаваемые данные.

8.1. Шифрование и дешифрование данных

Не секрет, что концепция шифрования данных возникла тысячи лет назад. Например, римские военачальники шифровали послания друг для друга с помощью кода Цезаря. Такой тип шифрования является одним из самых известных. Его сущность заключается в сдвиге символов в сообщении (в обычном тексте) на фиксированное число позиций в алфавите (так получается зашифрованный текст).

Затем зашифрованное сообщение может быть направлено к получателю любым, даже самым небезопасным способом, например курьером. Курьер не сможет прочитать текст сообщения, поскольку для человека, не знакомого с таким типом шифрования, он будет представлять собой не более чем набор случайных символов.

Восстановить исходное сообщение мог только получатель. Для этого он просто сдвигал все символы обратно.

Ниже представлены примеры обычного и зашифрованного текстов (со сдвигом на единицу):

```
обычный текст: "Иди к реке";
зашифрованный текст: "Йей л сёлё".
```

Процесс преобразования простого текста в шифр называется «шифрование», а обратный процесс – «дешифрование».

Кодом Цезаря люди пользовались около 150 лет назад, но для современного мира он слишком прост. Новейшие методы шифрования представляют собой более сложные алгоритмы. Однако все они выстраиваются по принципу секретного ключа, доступ к которому имеют две стороны.

8.2. Два типа ключей шифрования

Традиционный способ шифрования передаваемых данных – использование секретного ключа коллективного пользования (пароля, словесного выражения, числа), который знали бы только отправляющая

и получающая стороны. Дешифровать сообщение без такого ключа не представляется возможным.

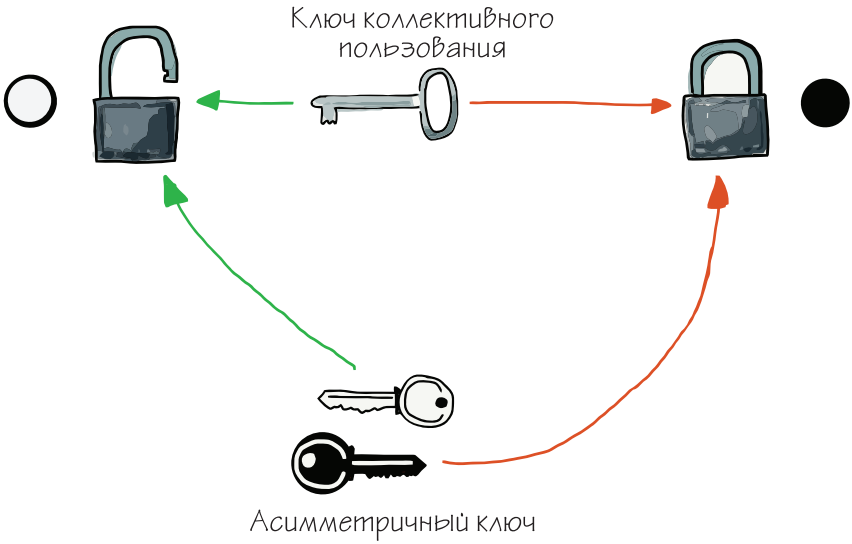

Рис.8.1. Ключ коллективного пользования vs асимметричный ключ

На ранних этапах развития интернета, перед тем как отправить зашифрованное сообщение, один человек звонил другому по телефону и сообщал ему ключ шифрования. Такой способ мог быть применим только для нескольких пользователей, но по мере увеличения их числа он становился все менее эффективным. Например, компания может иметь миллионы клиентов, и позвонить каждому из них просто не представляется возможным.

Передавать ключи шифрования коллективного пользования через интернет также небезопасно, поскольку злоумышленники могут отследить и перехватить незашифрованное сообщение, содержащее ключ. Также злоумышленники могут перехватить сообщение, дешифровать его, изменить содержимое и зашифровать повторно, а после чего отправить его на конечное устройство. Оно в свою очередь дешифрует сообщение и не узнает, какие именно изменения внесли злоумышленники.

Таким образом, ключи шифрования коллективного пользователя не подходят для защиты трафика в условиях существования триллионов пар устройств, взаимодействующих по сети.

В 1970-х годах была разработана концепция асимметричного ключа шифрования. Ее сущность заключается в том, что один ключ используется для шифрования сообщения, а другой – для его дешифрования. Устройство, которое будет получать зашифрованные данные, выбирает ключ шифрования и ключ дешифрования. Затем ключ шифрования отправляется на устройство, которое будет отправлять данные. Оно в свою очередь зашифровывает данные и отправляет их. Далее принимающее устройство восстанавливает исходное сообщение с помощью ключа дешифрования.

Так, согласно этой концепции, ключ шифрования называют открытым, так как он может передаваться другим устройствам, а ключ дешифрования – закрытым, поскольку он не может быть передан другому устройству. Очень часто такой тип называют шифрованием с открытыми/закрытыми ключами.

Благодаря разработке такой технологии злоумышленники могут получить только открытый ключ (как правило, в незашифрованном виде) и зашифрованный текст, который невозможно дешифровать без закрытого ключа. Угадать закрытый ключ также практически невозможно, поскольку он зашифровывается с помощью математических операций с большими простыми числами.

Итак, с появлением технологии открытых/закрытых ключей оставался единственный вопрос: как реализовать ее в сетевой модели.

8.3. Слой защищенных сокетов (SSL)

Поскольку проблемы безопасности вышли на первый план почти через 20 лет после разработки интернет-протоколов, при их решении для инженеров было крайне важно не нарушить существующую интернет-архитектуру. Поэтому они решили добавить дополнительный уровень между транспортным и прикладным. Они назвали его «слой защищенных сокетов» (Secure Sockets Layer – SSL), или «протокол защиты транспортного уровня» (Transport Layer Security – TLS).

Таким образом, когда приложение запрашивало транспортный уровень установить соединение с удаленным хостом, оно также могло запросить, чтобы соединение было зашифрованным или незашифрованным. Если было запрошено зашифрованное соединение, транспортный уровень зашифровывал данные перед разбивкой потока на пакеты. Таким образом, этот процесс не влиял на работу транспортного, сетевого и физических (канальных) и прикладного уровней.

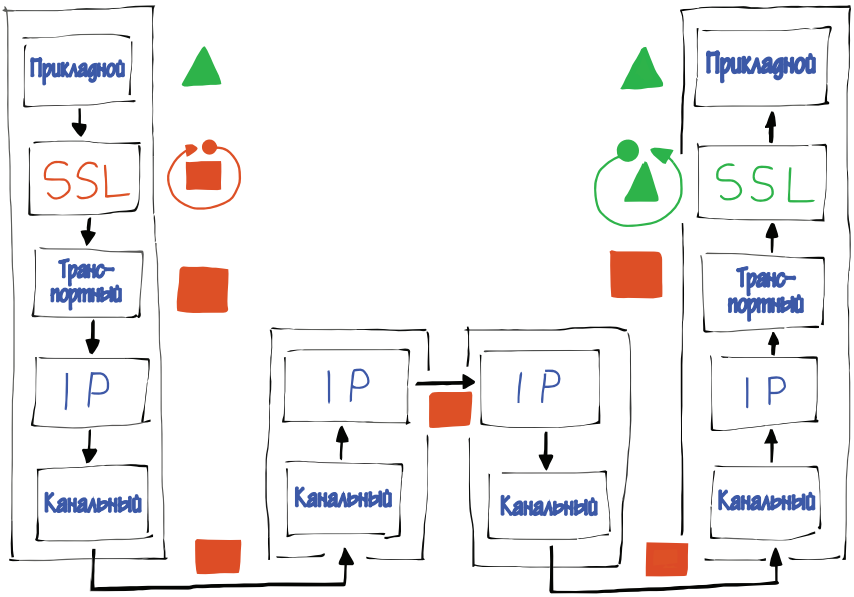

Рис.8.2. Где происходит шифрование и дешифрование

Шифрование представляло собой простое и прозрачное дополнение к транспортному уровню, благодаря чему не возникало необходимости в замене маршрутизаторов сетевого и канального уровней. Также для обеспечения безопасности не пришлось менять оборудование канального уровня. На прикладном же уровне единственным изменением было внедрение возможности осуществления приложением запроса на шифрование (при необходимости).

8.4. Шифрование трафика веб-браузера

Веб-браузеры и веб-серверы работают на прикладном уровне, поэтому пользователи зачастую даже не обращают внимание на то, какие соединения они используют, – зашифрованные или незашифрованные. Для того чтобы указать, что взаимодействие с веб-сервером должно проходить по зашифрованному соединению, веб-браузер меняет префикс «http:» на «https:». После этого в окне браузера появляется значок замочка, который сообщает пользователю, что веб-сайт, на который он переходит, защищен.

Установка https-соединений, шифрование и дешифрование данных всегда сопряжены с дополнительными накладными расходами,

ввиду чего какое-то время https-соединения использовались только для страниц, содержащих пароли, номера банковских счетов или другие конфиденциальные данные.

Однако со временем, по мере роста скорости сетей и повышения эффективности реализаций https, шифрование стало применяться всякий раз, когда пользователь взаимодействовал с веб-сервером, на котором имелась его учетная запись. Сегодня же наблюдается тенденция к использованию https-соединений для всего веб-трафика.

8.5. Сертификаты безопасности и центры сертификации

Несмотря на то что асимметричное шифрование позволяет передавать ключи шифрования по небезопасным сетям и использовать эти ключи для шифрования передаваемых данных, существует еще одна проблема безопасности. Она заключается в достоверности открытых ключей шифрования.

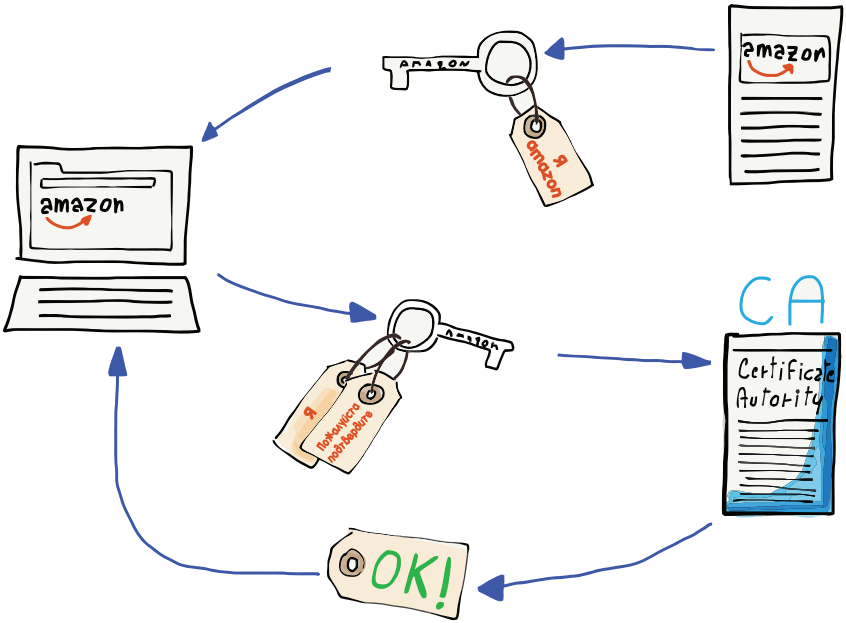

Рис.8.3. Центры сертификации и открытые ключи

К примеру, пользователь может быть уверен, что заходит на сайт **www.amazon.com**, но злоумышленники могут перехватить трафик и вы-

дать за нужный вам сайт любой другой, предоставив при этом вашему устройству открытый ключ шифрования. Так, ваш веб-браузер будет шифровать информацию о ваших банковских счетах с помощью ложного открытого ключа и отправит информацию прямо в руки мошенникам. И, поскольку злоумышленники предоставили открытый ключ, они также имеют и закрытый, что позволит им расшифровать вашу конфиденциальную информацию.

Так, вашему устройству необходимы гарантии того, что ключи являются достоверными. Достоверные ключи всегда подписываются цифровой подписью *центра сертификации* (Certificate Authority – CA). При соответствующей настройке на вашем устройстве уже будет информация о ряде широко известных центров сертификации. Поэтому, если устройству попадется ключ, подписанный одним из таких центров, оно доверится ему и будет использовать его для шифрования и передачи данных. Если же ваше устройство получит ключ, не имеющий соответствующей подписи, перед отправкой данных с использованием такого ключа оно уведомит вас о потенциальной угрозе безопасности.

Если вы видите такое уведомление, лучше всего отказаться от передачи данных по этому соединению, а также выяснить причину возникшей проблемы.

8.6. Заключение

Поскольку на момент того, как обострились вопросы безопасности данных, интернет существовал уже около 20 лет, главной задачей инженеров было обеспечить защиту данных, не нарушив уже устоявшуюся четырехуровневую модель. Лучшим решением было сделать безопасность дополнением к транспортному уровню. Именно поэтому протоколы, обеспечивающие безопасную передачу данных, называются «слой защищенных сокетов» (SSL) и «протокол защиты транспортного уровня» (TLS).

Создание асимметричного шифрования позволило решить проблему передачи ключей, существовавшую в подходах, где применялось шифрование ключей коллективного использования. Благодаря асимметричному шифрованию открытые ключи могут передаваться через небезопасные носители. Таким образом, можно установить безопасное соединение через небезопасное.

Обеспечение безопасности транспортного уровня никак не повлияло на работу сетевого, канального и прикладного уровней. Данный подход гарантирует, что все данные, передаваемые через соединения,

будут зашифрованы, перед тем как покинут ваше устройство. Учитывая популярность беспроводных соединений, таких как Wi-Fi, которые подвержены атакам злоумышленников, при работе с ними рекомендуется использовать исключительно защищенные соединения.

Заменяя префикс URL-адреса «http:» на «https:», веб-браузеры переходят на использование защищенного соединения. К слову, по префиксу пользователи могут определить тип соединения. Достоверность ключей шифрования таких соединений гарантируется центрами сертификации, подтверждающими, что ключ действительно принадлежит той организации, на чей веб-сайт вы пытаетесь зайти.

Алгоритм безопасности транспортного уровня представляет собой надежный и в то же время простой механизм, позволяющий защищать данные в масштабе триллионов пар взаимодействующих устройств.

8.7. Глоссарий

Асимметричное шифрование – подход к шифрованию, при котором один (открытый) ключ используется для шифрования данных перед передачей, а другой (закрытый) ключ используется для дешифрования данных после их получения.

Центр сертификации – организация, проверяющая достоверность открытых ключей безопасности и дающая им цифровую подпись.

Зашифрованный текст – закодированная версия сообщения, которую нельзя прочитать, не зная ключа и техники дешифрования.

Дешифрование – преобразование зашифрованного сообщения в обычное текстовое сообщение с использованием ключа безопасности.

Шифрование – преобразование обычного текстового сообщения в зашифрованное с ключом безопасности.

Обычный текст – сообщение, которое будет зашифровано перед отправкой.

Закрытый ключ – часть пары ключей, которая используется для дешифрования данных.

Открытый ключ – часть пары ключей, которая используется для шифрования данных.

Секретный ключ коллективного использования – подход к шифрованию, подразумевающий использование одного и того же ключа для шифрования и дешифрования.

Secure Sockets Layer (SSL) – слой защищенных сокетов. Протокол, который позволяет приложению запрашивать шифрование соедине-

ния транспортного уровня при прохождении через сеть. Является предшественником протокола транспортного уровня (TLS).

Transport Layer Security (TLS) – протокол, который позволяет приложению запрашивать шифрование соединения транспортного уровня при прохождении через сеть. Является последователем слоя защищенных сокетов (SSL).

8.8. Контрольные вопросы

Вы можете пройти этот тест онлайн по адресу **http://www.net-intro.com/quiz/**.

1. Как в веб-браузере указать, что мы хотим установить безопасное соединение?
   А. Использовать в URL-адресе префикс «https://».
   Б. Использовать безопасный веб-браузер.
   В. Открыть окно в режиме инкогнито.
   Г. Закодировать адрес сервера вручную с помощью SHA1.

2. Почему шифрование веб-трафика посредством подхода с ключом безопасности коллективного доступа неэффективно?
   А. Люди могут потерять ключ.
   Б. Ключи сложно передавать.
   В. Алгоритм шифрования и дешифрования данных с помощью одного и того же ключа очень уязвим.
   Г. Алгоритм шифрования и дешифрования данных с помощью одного и того же ключа требует большей вычислительной мощности.

3. Какая математическая концепция лежит в основе надежного асимметричного шифрования?
   А. Непрерывные функции.
   Б. Ряд Тейлора.
   В. Карта Карно.
   Г. Простые числа.

4. Какой ключ может быть передан по сети в виде обычного текста без ущерба для безопасности?
   А. Ключ шифрования.
   Б. Ключ дешифрования.
   В. Ключ безопасности коллективного доступа.
   Г. Универсальный ключ безопасности.

5. Где в четырехуровневой интернет-архитектуре находится слой защищенных сокетов (SSL)?
   А. Под канальным уровнем.
   Б. Между канальным и сетевым уровнями.
   В. Между сетевым и транспортным уровнями.
   Г. Между транспортным и прикладным уровнями.

6. Предположим, что вы подключились к Wi-Fi в каком-либо кафе и пользовались исключительно https-соединениями. Что из перечисленного может представлять угрозу потери данных кредитной карты при совершении онлайн-покупки в описанных выше условиях?
   А. Кто-то перехватит пакеты при их переходе по сети Wi-Fi.
   Б. Кто-то перехватит пакеты в маршрутизаторы шлюза.
   В. Кто-то перехватит пакеты в магистральном маршрутизаторе.
   Г. На вашем устройстве окажется вирус, который перехватывает нажатия клавиш.

7. Где происходит шифрование и дешифрование пакетов при использовании SSL?
   А. Они зашифровываются и дешифруются по мере прохождения через маршрутизатор.
   Б. Каждый физический канал шифруется в отдельности.
   В. Они зашифровываются на исходном устройстве и дешифруются на сервере.
   Г. Они зашифровываются в шлюзе Wi-Fi и дешифруются на последнем маршрутизаторе перед конечным устройством.

8. Какие изменения потребовалось внести в четырехуровневую модель интернет-архитектуры при внедрении SSL?
   А. Изменений не потребовалось.
   Б. Пришлось добавить поддержку Secure IP (IPSEC).
   В. Пришлось добавить возможность работы с более длинными пакетами.
   Г. Потребовалась возможность шифрования TTL.

9. Предположим, что в целях обеспечения защиты ваших данных вы используете исключительно асимметричное шифрование и ваши данные проходят по подводному кабелю. Какие из перечисленных ниже данных злоумышленникам было бы сложнее всего заполучить в описанных условиях?

А. С какими серверами вы общались.
Б. Как часто вы общались с серверами.
В. Объем данных, которые вы получили с серверов.
Г. Какие данные вы получили с серверов.

10. Какова роль центров сертификации в алгоритме асимметричного шифрования?

А. Они гарантируют, что ученики не смогут подделать значки для учебных занятий.
Б. Они гарантируют, что пакеты направляются в нужное место.
В. Они гарантируют достоверность открытого ключа.
Г. Они определяют переход определенных стран с IPv4 на IPv6.

11. Сеть ARPANET появилась в 1960-х годах. SSL же появился только в 1980-х годах. Каким образом в ARPANET решалась проблема безопасности данных при подключении к новой сети?

А. Все данные шифровались с помощью асимметричного шифрования.
Б. Шифрование осуществлялось на канальном уровне.
В. Существовала гарантия того, что никто не сможет получить доступ к физическим каналам.
Г. Использовались только безопасные маршрутизаторы Wi-Fi.

12. Что из перечисленного является фразой «безопасность – это весело», зашифрованной с помощью кода Цезаря со сдвигом в одну позицию?

А. Злохцжшфхшщг – дщх илшлтх.
Б. Йнрчшиъцчъые – ёыч кнънфч.
В. Вёипрбтоптуэ – юуп гётёмп.
Г. Цъэдехжгджзс – тзд чъжъбд.

13. Какой сдвиг используется в зашифрованной с помощью кода Цезаря фразе «щъс ъэмохяюл нсфыьмюъыюяи»?

А. 11.
Б. 1.
В. 20.
Г. 25.

Глава 9

Сетевая модель OSI

Во всех предыдущих разделах мы рассматривали трехуровневую модель, используемую для разработки и реализации протоколов TCP/IP и приложений. Однако TCP/IP не единственная модель, которая может нам помочь разобраться в строении интернет-архитектуры. Другая модель называется *моделью взаимодействия открытых систем* (The Open Systems Interconnection model – OSI). В отличие от TCP/IP, разрабатываемой и развивающейся по мере развертывания и изменения протоколов TCP/IP, модель OSI является результатом тщательной работы экспертов, занимающихся в том числе общим подходом к сетевым моделям.

В современной сети модели OSI и TCP/IP служат двум разным целям[1]. Модель TCP/IP – это модель реализации, поскольку она предоставляет руководство для тех, кто будет создавать сетевое оборудование или программное обеспечение, совместимое с TCP/IP. Модель OSI – это, скорее, абстрактная модель, которую можно использовать для понимания широкого спектра сетевых архитектур.

Хотя TCP/IP является наиболее широко используемой сетевой технологией на сегодняшний день, за последние 50 лет было реализовано и развернуто множество различных типов сетей. По мере их развития и совершенствования могут появиться новые модели реализации.

Модель OSI имеет семь уровней. Начиная с нижней части (ближайшей к физическим соединениям), модели OSI содержит следующие уровни: (1) физический, (2) канальный, (3) сетевой, (4) транспортный, (5) сеансовый, (6) представления и (7) прикладной. Мы рассмотрим каждый уровень модели OSI по очереди, начиная с физического уровня.

---

[1] Разумеется, такой подход слишком упрощен. До 1990 года существовали реализации операционных сетей, основанные на спецификациях ISO, которые точно следовали сетевой модели OSI. Однако сегодня реализации сетей ISO/OSI больше не используются повсеместно.

9.1. Физический (первый уровень)

Физический уровень OSI отвечает за физические атрибуты проводного, беспроводного, оптоволоконного или другого соединения, которое используется для передачи данных по одному каналу. Физический уровень также определяет формы разъемов и тип носителя, который можно использовать. Также этот уровень разбирается с тем, как кодировать биты (нули и единицы), из которых состоят данные, передаваемые через сетевую среду[2]. «Битовое кодирование» (или модуляция) определяет, насколько быстро данные могут быть отправлены по каналу.

9.2. Канальный (второй уровень)

Канальный уровень OSI отвечает за то, как системы, использующие физический канал, взаимодействуют друг с другом. Когда данные разбиваются на пакеты, канальный уровень определяет последовательности для обозначения начала и конца каждого пакета. Станциям, обменивающимся данными с использованием физического соединения, назначаются адреса, чтобы обеспечить эффективное использование носителя. Иногда несколько станций используют один и тот же носитель (как в беспроводной сети), и канальный уровень определяет, как эти станции будут совместно использовать соединения с другими системами, подключенными к сети. В большинстве случаев канальный уровень ориентируется на некоторую контрольную сумму для обнаружения и/или исправления ошибок в передаваемых данных.

Проблемы физического и канального уровней модели OSI решаются на канальном уровне модели TCP/IP.

9.3. Сетевой (третий уровень)

Подобно сетевому (IP) уровню в модели TCP/IP, сетевой уровень OSI занимается глобальным назначением «маршрутизируемых» адресов различным системам, подключенным к сети. Сетевой уровень управляет тем, как маршрутизаторы транспортируют пакеты от исходного к конечному устройству. Как и сетевой уровень TCP/IP, сетевой уровень OSI не занимается устранением ошибок, поскольку это относится к задачам следующего уровня.

---

[2] Один из наиболее распространенных методов кодирования битов для передачи по проводным соединениям – «манчестерское кодирование».

9.4. Транспортный (четвертый уровень)

Транспортный уровень в модели OSI отвечает за потерянные пакеты и их повторную передачу, а также занимается управлением потоками и контролирует размер окна. Остальные функции транспортного уровня TCP/IP выполняются на следующем уровне в модели OSI.

9.5. Сеансовый (пятый уровень)

Сеансовый уровень OSI обрабатывает установку соединений между приложениями. Сеансовый уровень работает с портами, и его задача помочь клиентскому приложению найти правильное серверное приложение в конкретной системе. Также на данном уровне модели OSI решаются некоторые вопросы безопасности.

9.6. Уровень представления (шестой уровень)

Уровень представления контролирует представление и кодировку данных для передачи их по сети. Например, он отвечает за то, как кодировать пиксели изображения, чтобы принимающее приложение могло правильно декодировать данные. Уровень представления также обрабатывает шифрование и дешифрование данных.

9.7. Прикладной (седьмой уровень)

Прикладной уровень OSI очень похож на прикладной уровень модели TCP/IP, поскольку на этом уровне также работают приложения. Здесь также существуют клиентские приложения, которые инициируют подключения, и серверные, которые отвечают на запросы подключения. Различные пары приложений имеют стандарты протоколов, которые определяют взаимодействие между несколькими клиентами и несколькими серверами от разных поставщиков.

9.8. Сравнение моделей OSI и TCP/IP

Мы можем использовать модель OSI для обеспечения альтернативного представления модели TCP/IP. Так, мы можем сравнить, как модель OSI разбивает функциональность сети на свои уровни и как модель TCP/IP нарушает ее функциональность.

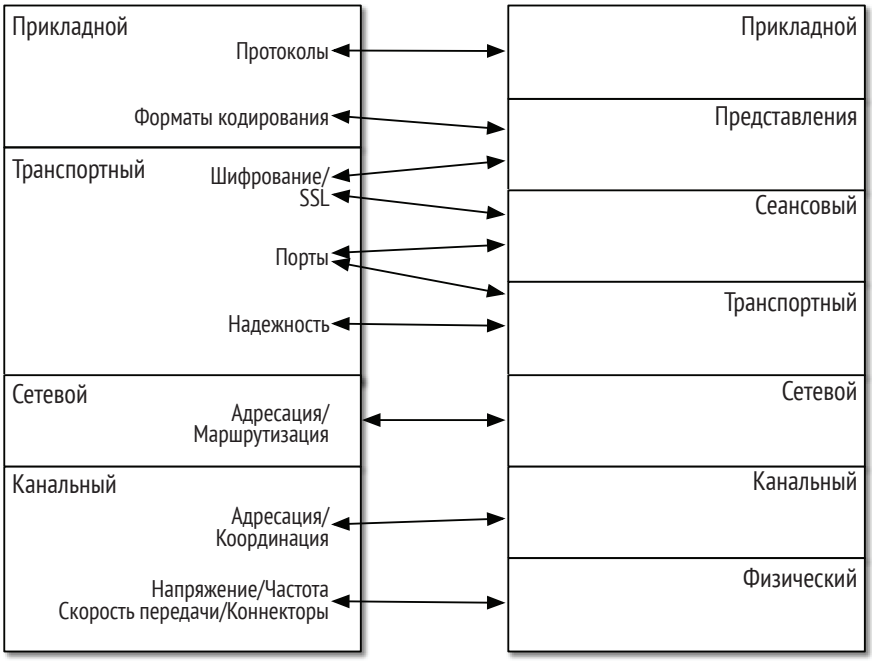

Рис.9.1. Сравнение моделей TCP и OSI

9.9. Канальный уровень (TCP/IP)

Данный уровень TCP/IP объединяет физический и канальный уровни из модели OSI. Как правило, эти уровни реализуются аппаратно. Такие технологии, как Ethernet, Wi-Fi, спутниковая связь или оптоволокно, часто реализуются в карте сетевого драйвера, которая подключается к задней части компьютера или маршрутизатора. Карта сетевого драйвера обычно реализует как физический, так и канальный аспекты соединения в аппаратном обеспечении карты. В большинстве случаев канальный уровень настроен на ограничения и требования соответствующих физических уровней. Таким образом, в реальных системах довольно редко конкретный канальный уровень может быть произвольно соединен с любым количеством физических уровней. Поскольку может быть трудно разделить физические аспекты и аспекты канала передачи данных для конкретной технологии связи, модель TCP объединяет их в один уровень, что значительно упрощает задачу.

9.10. Сетевой уровень (TCP/IP)

Единственный аспект, совпадающий у двух моделей практически полностью, – сетевые уровни OSI и TCP/IP. Они выполняют одинаковые функции – создают глобально маршрутизируемое адресное пространство и контролируют маршрутизаторы, обеспечивая оптимальную транспортировку пакетов.

9.11. Транспортный уровень (TCP/IP)

Функции транспортного уровня в TCP/IP распределены в модели OSI между транспортным и сеансовым уровнями. Транспортный уровень OSI занимается управлением потоком и повторной передачей пакетов, в то время как уровень представления OSI взаимодействует с приложениями, работающими на нескольких портах, а также решает проблемы разрыва сеанса. Слой защищенных сокетов (SSL) в модели TCP/IP соответствует сеансовому уровню и уровню представления в модели OSI.

9.12. Прикладной уровень (TCP/IP)

Прикладной уровень TCP/IP объединяет не связанные с безопасностью аспекты уровня представления и прикладного уровня OSI. Многие приложения модели TCP/IP решают такие проблемы, как кодирование и декодирование различных типов данных, данная модель не имеет отдельного уровня, отвечающего за форматирование данных. В приложениях TCP/IP используются различные технологии кодирования и декодирования данных, но TCP/IP зачастую рассматривает их как библиотечный код, который приложения используют по мере необходимости.

9.13. Заключение

Хотя модель TCP/IP, описанная в этой книге, широко используется для руководства реализацией сетей TCP/IP, оборудования и программного обеспечения, модель OSI может помочь нам взглянуть и сравнить широкий спектр сетевых архитектур, начиная от открытых сетей, заканчивая частными сетями конкретных производителей.

9.14. Глоссарий

Абстрактная модель – модель и набор терминов, которые используются для общего понимания проблемной области и руководства разработкой стандартов и реализаций для решения проблем.

Модель реализации – модель и набор терминов, которые используются для руководства при разработке стандартов и реализации для решения конкретной проблемы.

ISO (International Organization for Standardization) – международная организация по стандартизации. Всемирная организация, разрабатывающая стандарты в области вычислений, сетей и многих других областях.

OSI (The Open Systems Interconnection model) – модель взаимодействия открытых систем. Семиуровневая модель, используемая в целях организации разработки различных подходов к сетевой архитектуре.

9.15. Контрольные вопросы

Вы можете пройти этот тест онлайн по адресу **http://www.net-intro.com/quiz/**.

1. В чем состоит особенность сетевой модели OSI?
   А. Сети OSI используются в южном полушарии.
   Б. Подход OSI можно использовать для анализа множества различных сетевых моделей.
   В. Сети OSI лучше используют ограниченную полосу пропускания.
   Г. Сети OSI более безопасны.
2. Сколько уровней у модели OSI?
   А. Четыре.
   Б. Шесть.
   В. Семь.
   Г. Девять.
3. Какой из уровней OSI определяет форму разъемов для сетевых подключений?
   А. Физический.
   Б. Канальный.
   В. Сетевой.
   Г. Транспортный.
4. Какой из уровней наиболее схож между сетевыми моделями OSI и TCP?
   А. Канальный уровень TCP и канальный уровень OSI.
   Б. Сетевой уровень TCP и сетевой уровень OSI.

В. Транспортный уровень TCP и транспортный уровень OSI.
Г. Прикладной уровень TCP и сеансовый уровень OSI.

5. Какому уровню соответствует слой защищенных сокетов (SSL) модели TCP/IP в модели OSI?
   А. Уровень защищенного канала данных (SDLL).
   Б. Безопасный сетевой уровень (SNL).
   В. Безопасный транспортный уровень (STL).
   Г. Уровни сеансовый и представления.

6. Почему модель TCP объединяет канальный и физический уровни OSI в один канальный уровень?
   А. Модель TCP не имеет физического уровня.
   Б. Разработчики модели TCP были проигнорированы на собрании OSI 1981 года в Утрехте, Нидерланды.
   В. Довольно часто разработка канального и физического уровней тесно связаны .
   Г. Разработчики хотели сделать модель TCP более понятной для пользователей.

Глава 10

Заключительная часть

Как уже было сказано, создание интернета решило самую сложную на сегодняшний день инженерную проблему в мире. Его разработка началась более 50 лет назад. На протяжении последних 50 лет интернет непрерывно совершенствовался, и, безусловно, он продолжит развиваться в будущем.

Сегодня интернет объединяет миллиарды устройств по всему миру с помощью огромного количества маршрутизаторов и канальных соединений. Он настолько сложен, что никогда не работает в полную силу. Особенность интернета заключается не в его «идеальности», а в том, насколько легко он может адаптироваться к проблемам, сбоям в работе, ошибкам, потере данных и многим другим непредвиденным проблемам. Его отличительные черты – гибкость и способность решать любые возникающие проблемы.

Рис.10.1. Четырехуровневая модель

Для того чтобы создать столь масштабную систему, инженеры разбили ее на четыре отдельных уровня.

- Канальный/физический уровень отвечает за всю сложную инженерную работу, благодаря которой данные могут осуществить

переход с помощью различных технологий, таких как беспроводная сеть Wi-Fi, проводной Ethernet, оптоволоконные или спутниковые соединения.

- Сетевой (IP) уровень отвечает за маршрутизацию данных в условиях серии переходов. Он обеспечивает быструю и эффективную транспортировку данных от одного из миллиарда исходных устройств к любому из миллиардов конечных устройств. Сетевой уровень динамически корректирует и перенаправляет данные в зависимости от сетевой нагрузки, производительности канала или возникающих сбоев. Несмотря на эффективность этого уровня, иногда он может терять или даже отбрасывать данные. Сетевой уровень отвечает не за обеспечение общей надежности сети, а за обеспечение оптимальной маршрутизации и транспортировки данных.
- Задача транспортного уровня – компенсировать недостатки сетевого и канального уровней. Он обеспечивает повторную передачу потерянных пакетов, а также упорядочение пакетов, пришедших не по порядку, перед их передачей в принимающее приложение. Также транспортный уровень отвечает за управление потоками между отправляющим и принимающим приложениями. Таким образом, он гарантирует, что при высокоскоростном и неперегруженном соединении данные будут перемещаться быстро, а при соединении с низкой скоростью или загруженном соединении – медленно. Посредством регулирования потока данных и скорости передачи данных транспортный уровень обеспечивает бесперебойную работу интернета даже при больших нагрузках.
- Три перечисленных уровня призваны облегчать работу прикладного уровня. Благодаря этому приложения могут устанавливать сетевое соединение и отправлять/получать данные по этому соединению с помощью всего нескольких строк кода. Таким образом, приложения могут сосредоточиться на решении пользовательских задач. Стоит отметить, что благодаря вышеописанному подходу на сегодняшний день мы можем наблюдать широкий спектр самых разных приложений, для внедрения которых не требуются изменение основных интернет-протоколов.

Без разделения интернета на уровни было бы намного сложнее построить и развернуть постоянно совершенствующиеся версии се-

ти. Если бы каждому приложению приходилось решать задачи всех четырех уровней, богатство и разнообразие сетевых приложений бы значительно сократилось.

Прогресс, достигнутый за 50 лет постоянного совершенствования интернета, действительно удивляет. Но относительно создания сетевых приложений мы все еще находимся в начале пути. Нетрудно представить себе интернет, в котором каждый выключатель, лампочка, холодильник, стол, автомобиль, дорога, дрон и стул имеют свой интернет-адрес, и всем им необходимо взаимодействовать друг с другом. Таким образом, на горизонте возникают новые инженерные задачи, для решения которых, возможно, потребуется добавить в существующую интернет-архитектуру новые уровни.

Однако, если у инженеров прошлого получилось разработать сетевые протоколы, позволяющие взаимодействовать между собой даже не сотням, а миллиардам устройств, подключенных к сети, то нынешние и будущие инженеры обязательно справятся со всеми проблемами, возникающими на их пути к обеспечению взаимодействия между собой триллионов устройств.

Книги издательства «ДМК ПРЕСС» можно купить оптом и в розницу
в книготорговой компании «Галактика»
(представляет интересы издательств
«ДМК ПРЕСС», «СОЛОН ПРЕСС», «КТК Галактика»).

Адрес: г. Москва, пр. Андропова, 38;
Тел.: **+7(499) 782-38-89**. Электронная почта: **books@alians-kniga.ru**.
При оформлении заказа следует указать
адрес (полностью), по которому должны быть высланы книги;
фамилию, имя и отчество получателя.
Желательно также указать свой телефон и электронный адрес.

Эти книги вы можете заказать и в интернет-магазине: **www.a-planeta.ru**.

Чарльз Р. Северанс

Как работают компьютерные сети и интернет

Главный редактор *Мовчан Д. А.*
dmkpress@gmail.com
Перевод с английского *Бомбакова П. М.*
Редактор *Яценков В. С.*
Корректор *Абросимова Л. А.*
Верстка *Паранская Н. В.*
Дизайн обложки *Мовчан А. Г.*

Формат 60×90 1/16.
Печать цифровая. Усл. печ. л. 7,25.
Тираж 200 экз.

Веб-сайт издательства: **www.dmkpress.com**