Министерство образования и науки Российской Федерации
НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Н.Л. ДОЛОЗОВ

КОМПЬЮТЕРНЫЕ СЕТИ

Учебно-методическое пособие

НОВОСИБИРСК
2013

УДК 004.7(075.8)
Д 648

Рецензенты:
д-р техн. наук, профессор *Б.Ю. Лемешко*
канд. техн. наук, доцент, директор ЦИУ *В.М. Стасышин*

Долозов Н.Л.
Д 648 Компьютерные сети: учеб.-метод. пособие / Н.Л. Долозов. – Новосибирск: Изд-во НГТУ, 2013. – 112 с.

ISBN 978-5-7782-2379-0

Учебно-методическое пособие посвящено курсу «Компьютерные сети» и служит руководством при выполнении лабораторных работ, проводимых со студентами III курса (направлений 010500.62 «Математическое обеспечение и администрирование информационных систем» и 010400.62 «Прикладная математика и информатика») в терминальных классах ФПМИ.

Работа подготовлена на кафедре программных систем и баз данных
и утверждена Редакционно-издательским советом университета
в качестве учебно-методического пособия

УДК 004.7(075.8)

ISBN 978-5-7782-2379-0

ОГЛАВЛЕНИЕ

ПРЕДИСЛОВИЕ

Компьютерные сети – это одна из самых важных и захватывающих технологий нашего времени. Два десятилетия назад доступ к сетям имело лишь ограниченное число пользователей. В настоящее время обмен данными между компьютерами стал неотъемлемой частью повседневной жизни. С осознанием важности компьютерных сетей и ростом их популярности появился устойчивый спрос на специалистов разных категорий, имеющих опыт работы с сетями. Компании все чаще привлекают специалистов, которые занимаются планированием, приобретением, установкой, эксплуатацией и управлением аппаратными и программными системами, лежащими в основе локальных и объединенных сетей. Современное компьютерное программирование не ограничивается задачами, которые решаются на отдельных компьютерах. Следовательно, программисты должны проектировать и реализовывать прикладное программное обеспечение, способное взаимодействовать с программным обеспечением, работающим на удаленных компьютерах.

В связи с этим студентам необходимо изучить и на практике освоить базовые средства для создания приложений, которые могли бы взаимодействовать в сети.

Цель настоящего учебно-методического пособия – помочь студентам (пользователям) теоретически и на практике, т. е. в формате лабораторных работ, освоить основные сетевые средства, начиная от самых низких уровней передачи данных и заканчивая наивысшими уровнями прикладного программного обеспечения.

Все лабораторные работы в пособии имеют одинаковую структуру, состоящую из трех частей. В первой части каждой лабораторной работы приводятся теоретические сведения и необходимые методические указания. Поскольку достаточно сложный и объемный теоретический материал представлен в краткой и понятной форме, эти сведения и

указания должны облегчить понимание студентами поставленных перед ними практических задач. Во второй части каждой работы сформулированы задания в различных вариантах. Третья часть каждой работы содержит контрольные вопросы, ответы на которые позволят студентам выделить в работе ключевые моменты и закрепить полученные знания и практические навыки. Кроме того, в пособии приведены список литературы и приложения, содержащие рисунки и примеры реализации некоторых программ, написанных на языке C++.

В заключение хотелось бы выразить благодарность студентам факультета прикладной математики и информатики, принимавшим участие в разработке программного обеспечения для реализации лабораторных работ.

Лабораторная работа № 1

АНАЛИЗ СТРУКТУРЫ ЛОКАЛЬНОЙ СЕТИ ФПМИ

Цель работы. Подготовить личную страничку бригады для размещения на ней отчетов по лабораторным работам. Выполнить анализ структуры локальной сети факультета ФПМИ и стека протоколов Internet.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ И МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Количество компьютеров, подключенных к глобальной сети Интернет, измеряется миллионами (и это число постоянно растет); таким образом, Интернет создает глобальную коммуникацию, позволяя огромному количеству пользователей обмениваться информацией и использовать вычислительные ресурсы друг друга. Кроме того, в настоящее время происходит тесная интеграция Интернета с мобильными и беспроводными технологиями, что значительно расширяет круг его функций. Интернет представляет собой всемирную компьютерную сеть, т. е. сеть, связывающую в единое целое миллионы вычислительных устройств, расположенных в разных уголках Земного шара.

Компьютеры, подключенные к сети, маршрутизаторы и другие «компоненты» Интернета используют протоколы, осуществляющие управление приемом и передачей информации внутри самого Интернета. Наиболее важными протоколами в глобальной сети являются TCP (Transmission Control Protocol – протокол управления передачей) и IP (Internet Protocol – интернет-протокол). Стек основных протоколов, использующихся в Интернете, известен под названием TCP/IP.

С точки зрения технологий и развития существование Интернета обеспечивается созданием, проверкой и внедрением интернет-стандартов. Эти стандарты вырабатываются проблемной группой разработок для Интернета (Internet Engineering Task Force, IETF). Документы, создаваемые IETF, носят название RFC (Requests For Comments – обращения за разъяснениями). Изначально подобные документы предназначались для разрешения архитектурных проблем, возникавших в сетях-предшественницах Интернета. Со временем ситуация сложилась так, что, формально не обладая статусом стандарта, документы RFC стали стандартами де-факто. В настоящее время эти документы составляются весьма точно и детально, описывая такие протоколы, как TCP, IP, HTTP (для web) и SMTP (для электронной почты). Существует более 6000 различных документов RFC.

Протокол определяет формат и очередность сообщений, которыми обмениваются два или более устройств, а также действия, выполняемые при передаче и/или приеме сообщений либо при наступлении иных событий. Протоколы очень широко используются как в компьютерных сетях вообще, так и в сети Интернет в частности.

ОБЩАЯ ИНФОРМАЦИЯ О ПРОТОКОЛАХ

Как известно, протоколы, а следовательно, и все сетевое программное и аппаратное обеспечение организованы в виде уровней. Каждый протокол относится к определенному уровню сетевой коммуникационной модели. Многоуровневая структура позволяет детально оценивать элементы большой и сложной системы, что уже является ее значительным достоинством. Кроме того, с использованием многоуровневой структуры легче модифицировать функции системы: нужно лишь внести изменения в соответствующий уровень, при этом структурно-функциональная организация системы останется прежней.

Совокупность протоколов всех уровней коммуникационной модели называется стеком протоколов.

Снабжение каждого уровня коммуникационной модели собственным протоколом наряду с достоинствами имеет и несколько недостатков. Первый связан с нередко встречающимся дублированием одних и тех же функций различными уровнями (например, контроль ошибок). Другой потенциальный недостаток заключается в том, что функциям одно-

го уровня может понадобиться информация, хранящаяся на другом уровне (например, время). Это нарушает принцип изолированности уровней многоуровневой структуры.

СТЕК ПРОТОКОЛОВ ИНТЕРНЕТА

Коммуникационная модель Интернета состоит из пяти уровней: физического, канального, сетевого, транспортного и прикладного. Вместо терминов «единица обмена сетевого уровня», «единица обмена канального уровня» используются специальные имена. Они приведены в следующей таблице.

Уровень	*Единица измерения*
Прикладной	Сообщение
Транспортный	Сегмент
Сетевой	Дейтаграмма/Пакет
Канальный	Кадр
Физический	Поток бит

Поддержка протоколов может быть аппаратной, программной или смешанной. Протоколы прикладного уровня, такие как HTTP и SMTP, а также протоколы транспортного уровня практически всегда поддерживаются программно. Напротив, протоколы физического и канального уровней, тесно связанные со средой передачи данных, поддерживаются аппаратно сетевой интерфейсной картой. Сетевой уровень, находящийся в центре коммуникационной модели, может поддерживаться как аппаратно, так и программно. Далее даны характеристики каждого из пяти уровней коммуникационной модели Интернета.

Прикладной уровень, как следует из его названия, предназначен для поддержки сетевых приложений. Имеется множество протоколов прикладного уровня, из которых наиболее важные HTTP (для путешествий по web-страницам), SMTP (для электронной почты) и FTP (для обмена файлами).

Главная функция **транспортного уровня** заключается в передаче сообщений прикладного уровня между клиентом и сервером. В Интер-

нете существуют два транспортных протокола: TCP и UDP. Протокол TCP обеспечивает передачу с установлением логического соединения, т. е. надежную передачу с контролем перегрузки. Протокол UDP обеспечивает передачу сообщений без установления логического соединения, т. е. ненадежный вид связи, где допускаются искажения и потери данных.

Сетевой уровень обеспечивает передачу дейтаграмм между двумя хостами и базируется на двух основных протоколах. Первый протокол определяет поля дейтаграммы и интерпретацию их содержимого маршрутизаторами и оконечными системами. Этот протокол – единственный протокол сетевого уровня в Интернете, он имеет название IP. Вторым протоколом является один из многочисленных протоколов маршрутизации, предназначенных для определения путей дейтаграмм от отправителя до адресата. Число протоколов маршрутизации огромно. Несмотря на функциональные различия между протоколом IP и протоколами маршрутизации, а также на широкое разнообразие последних, их обычно объединяют под общим именем IP, подчеркивая этим их связующую роль в организации глобальной Сети.

Протокол транспортного уровня (TCP или UDP) передает сегмент и адрес назначения протоколу IP сетевого уровня, подобно тому как вы опускаете письмо в почтовый ящик, а протокол IP сетевого уровня доставляет сегмент конечному хосту и передает его обратно транспортному уровню.

Сетевой уровень обеспечивает передачу пакета через серию маршрутизаторов между оконечными системами. Для перемещения пакета (дейтаграммы) от одного узла к другому сетевой уровень прибегает к службам канального уровня. Таким образом, основная функция канального уровня заключается в передаче дейтаграмм между узлами на маршруте.

Канальный уровень использует специальный протокол, ориентированный на линию связи. Иногда протоколы канального уровня обеспечивают надежную передачу между узлами. Обратите внимание на различие надежной передачи на транспортном и канальном уровнях: протокол TCP гарантирует надежность на всем пути следования сообщения, а протокол канального уровня – лишь между парой узлов. К протоколам канального уровня относятся Ethernet и PPP; иногда аналогичные функции несут технологии асинхронной передачи данных (ATM) и ретрансляции кадров. Поскольку путь от отправителя

до адресата обычно состоит из цепочки разнородных линий связи, передача дейтаграммы может осуществляться различными канальными протоколами.

Если назначением канального уровня является передача кадров между соседними узлами сети, то **физический уровень** обеспечивает передачу между узлами отдельных бит информации. Протоколы физического уровня также напрямую зависят от используемой линии связи (медной витой пары, одномодового оптоволокна и т. п.). Технология Ethernet поддерживает множество протоколов физического уровня, предназначенных для витой пары, коаксиального кабеля, оптоволоконного кабеля и некоторых других видов линий. В каждой из линий связи механизмы передачи бита различны.

СЕТЕВОЕ ОБОРУДОВАНИЕ

Основными компонентами сети являются рабочие станции, серверы, передающие среды (кабели) и сетевое оборудование. **Рабочими станциями** называются компьютеры сети, на которых пользователями сети реализуются прикладные задачи. **Серверы сети** – это аппаратно-программные системы, выполняющие функции управления распределением сетевых ресурсов общего доступа. Сервером может быть любой подключенный к сети компьютер, на котором находятся ресурсы, используемые другими устройствами локальной сети. В качестве аппаратной части сервера используются достаточно мощные компьютеры.

Выделяют следующие виды сетевого оборудования.

Сетевые карты – это контроллеры, подключаемые в слоты расширения материнской платы компьютера, предназначенные для передачи сигналов в сеть и приема сигналов из сети.

Терминаторы – это резисторы номиналом 50 Ом, которые обеспечивают затухание сигнала на концах сегмента сети.

Концентраторы (Hub) – это центральные устройства кабельной системы или сети физической топологии «звезда», которые при получении пакета на один из своих портов пересылают его на все остальные. В результате получается сеть с логической структурой общей шины. Различают концентраторы активные и пассивные. Активные концентраторы усиливают полученные сигналы и передают их. Пассивные концентраторы пропускают через себя сигнал, не усиливая и не восстанавливая его.

Повторители (репитер, от англ. repeater) – сетевое оборудование, предназначенное для увеличения расстояния сетевого соединения повторением электрического сигнала «один в один». Бывают однопортовые повторители и многопортовые. В терминах модели OSI повторитель работает на физическом уровне. Повторители не распознают MAC-адреса и поэтому не могут использоваться для уменьшения трафика.

Коммутаторы (Switch) – управляемые программным обеспечением центральные устройства кабельной системы, сокращающие сетевой трафик за счет того, что пришедший кадр анализируется для выяснения адреса его получателя и соответственно передается только ему.

Использование коммутаторов – более дорогое, но и более производительное решение. Коммутатор – обычно значительно более сложное устройство и может обслуживать одновременно несколько запросов. Если по какой-то причине нужный порт в данный момент времени занят, то пакет помещается в буферную память коммутатора, где и дожидается своей очереди. Построенные с помощью коммутаторов сети могут охватывать несколько сотен машин и иметь протяженность в несколько километров.

Маршрутизатор (Router) – стандартные устройства сети, работающие на сетевом уровне и позволяющие переадресовывать и маршрутизировать пакеты из одной сети в другую, а также фильтровать широковещательные сообщения.

Мосты (Bridge) – устройства сети, которое соединяют два отдельных сегмента, ограниченных своей физической длиной, и передают трафик между ними. Мосты также усиливают и конвертируют сигналы для кабеля другого типа. Это позволяет расширить максимальный размер сети, одновременно не нарушая ограничений на максимальную длину кабеля, количество подключенных устройств или количество повторителей на сетевой сегмент.

Шлюзы (Gateway) – программно-аппаратные комплексы, соединяющие разнородные сети или сетевые устройства. Шлюзы позволяют решать проблемы различия протоколов или систем адресации. Они действует на сеансовом, представительском и прикладном уровнях модели OSI.

Мультиплексоры – это устройства центрального офиса, которые поддерживают несколько сотен цифровых абонентских линий. Мультиплексоры посылают и получают абонентские данные по телефон-

ным линиям, концентрируя весь трафик в одном высокоскоростном канале для передачи в интернет или в сеть компании.

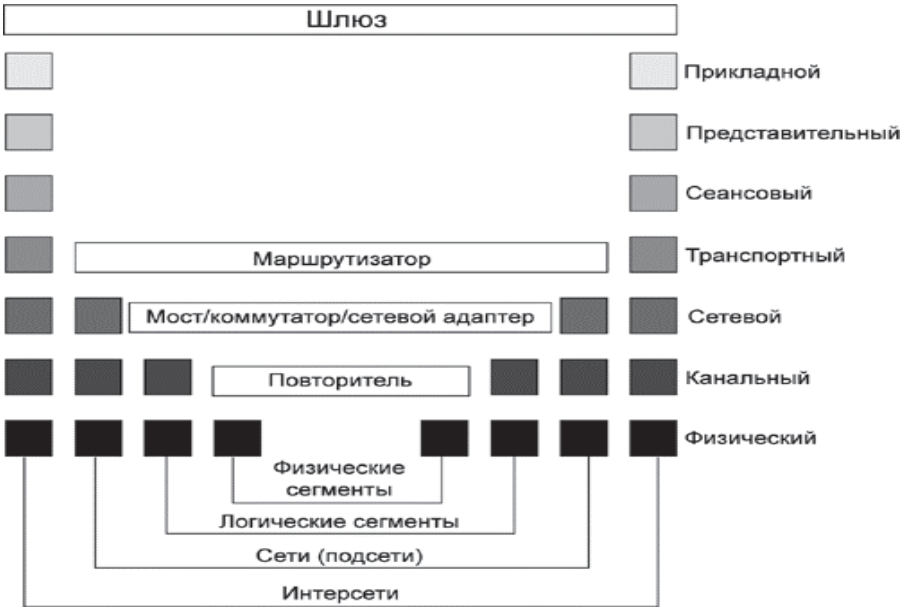


Рис. 1.1. Соответствие функций различных коммуникационных устройств уровням модели OSI

Межсетевые экраны (брандмауэры) – это сетевые устройства, реализующие контроль за поступающей в локальную сеть и выходящей из нее информацией и обеспечивающие защиту локальной сети посредством фильтрации информации.

На рис. 1.1 показана схема расположения основных коммуникационных устройств по уровням модели OSI.

ФИЗИЧЕСКАЯ СРЕДА ПЕРЕДАЧИ

Физические среды можно разделить на два типа: проводные и беспроводные. Проводные среды передачи предполагают наличие твердотельного проводника и включают оптоволоконный кабель, медную витую пару и коаксиальный кабель. В беспроводной среде

передача осуществляется без применения твердых проводников; этот тип среды используется в беспроводных локальных сетях и в спутниковой связи.

Медная витая пара – самый дешевый и наиболее популярный вид кабелей. На протяжении более чем 100 лет витая пара активно используется в телефонных сетях. Можно смело утверждать, что более 99 % всех кабелей, соединяющих абонентов с телефонными коммутаторами, представляют собой медные витые пары. Многие могли видеть эти кабели у себя дома или на работе. Витая пара состоит из двух изолированных медных проводов толщиной 1 мм, заключенных в спиральную оболочку. Внутри оболочки провода переплетены друг с другом, чтобы снизить уровень электрических помех, возникающих между парой проводников. Обычно перед помещением пар внутрь кабеля их снабжают дополнительными защитными экранами.

Неэкранированная витая пара (Unshielded Twisted Pair – UTP), как правило, используется в офисных локальных сетях, расположенных в одном здании. Скорость передачи данных в такой среде варьируется от 10 Мбит/с до 1 Гбит/с и определяется толщиной провода и расстоянием между обменивающимися сторонами. В локальных сетях обычно используется два типа неэкранированных витых пар: витая пара категории 3 и витая пара категории 5. Первая относится к голосовым линиям связи и характерна для офисов. Как правило, в офисах прокладывают две независимые витые пары: одну для телефонной связи, а другую для дополнительных телефонных соединений и локальной сети. В частности, в широко распространенной технологии Ethernet 10 Мбит/с применяют неэкранированную витую пару категории 3. Витая пара категории 5 имеет большее число витков на дюйм, а также снабжена тефлоновой изоляцией, что позволяет обеспечить более высокие скорости передачи данных. В последние годы получила распространение технология Ethernet 100 Мбит/с, в которой используется витая пара категории 5.

Коаксиальный кабель, как и витая пара, состоит из двух медных проводников, однако эти проводники, в отличие от витой пары, расположены не параллельно, а концентрически (коаксиально). С применением особых видов изоляции и экранирования коаксиальный кабель позволяет добиться более высоких скоростей передачи данных, чем витая пара. Коаксиальные кабели подразделяются на два вида: с немодулируемой передачей и с модулируемой передачей.

Коаксиальный кабель с немодулируемой передачей имеет сопротивление 50 Ом и толщину около 1 см; к его несомненным физическим достоинствам можно отнести легкость и гибкость. Этот тип кабеля часто применяется в локальных сетях наряду с неэкранированной витой парой. Термин «с немодулируемой передачей» означает, что битовый поток поступает в кабель без частотной модуляции. Хотя коаксиальный кабель может применяться в технологии Ethernet 10 Мбит/с, почти для всех новых реализаций Ethernet характерна неэкранированная витая пара.

Коаксиальный кабель с модулируемой передачей обладает сопротивлением 75 Ом и имеет бóльшую толщину, вес и меньшую гибкость по сравнению с кабелем с немодулируемой передачей. Часто кабель с модулируемой передачей используют в системах кабельного телевидения. Оба вида кабеля (с немодулируемой и с модулируемой передачей) относятся к классу разделяемых проводных сред передачи данных. Другими словами, информация, передаваемая или принимаемая одной системой, принимается всеми системами, подключенными к кабелю.

Оптоволоконная среда передачи представляет собой тонкий и гибкий кабель, внутри которого распространяются световые импульсы, несущие информацию о передаваемых битах. Даже простой оптоволоконный кабель способен передавать данные на огромных скоростях в десятки и даже сотни гигабит в секунду. Оптоволоконные линии не подвержены электрическим наводкам, имеют очень низкий уровень ослабления сигнала на единицу протяженности и обладают значительной устойчивостью к механическим воздействиям. Однако высокая стоимость оптических устройств (маршрутизаторов, приемников и передатчиков) делает нецелесообразным (по экономическим причинам) применение оптоволоконных линий связи для передачи на короткие расстояния, например в локальных офисных сетях или для резидентного домашнего доступа.

Радиоканалы передают сигналы с помощью электромагнитных волн радиодиапазона. Их достоинство заключается в том, что для связи не требуется твердотельного проводника сигналов (следовательно, нет необходимости в его прокладке), т. е. пользователь может быть мобильным, есть потенциал в увеличении расстояния передачи. Характеристики радиоканала зависят от среды передачи радиоволн и расстояния между

оконечными системами. К факторам среды передачи относятся затухание сигнала вследствие распространения в среде, прохождение через поглощающие предметы, взаимодействие с отраженными электромагнитными волнами, а также волнами, исходящими от других источников излучения.

Спутник связи организует взаимодействие двух или более наземных приемопередатчиков. Он принимает сигналы одного частотного диапазона, производит их регенерацию с помощью повторителя, а затем передает сигналы в другом частотном диапазоне. Скорость обмена данными, обеспечиваемая спутниковыми каналами, составляет несколько гигабит в секунду. Существуют два типа спутников: геостационарные и низкоорбитальные.

ПРИНЦИП ВЗАИМОДЕЙСТВИЯ ПРИКЛАДНОЙ ПРОГРАММЫ С СИСТЕМНЫМ ПРОГРАММНЫМ ОБЕСПЕЧЕНИЕМ

Для взаимодействия прикладных сетевых программ ОС предоставляет механизм сокетов (sockets). Механизм сокетов реализует удобный и достаточно универсальный интерфейс обмена сообщениями, предназначенный для разработки сетевых распределенных приложений. Его универсальность обеспечивают следующие концепции.

- Независимость от нижележащих сетевых протоколов и технологий.
- Использование абстрактной конечной точки соединения, получившей название сокет (socket – гнездо). Сокет – это точка, через которую сообщения уходят в сеть или принимаются из сети. Сетевое соединение между двумя процессами осуществляется через пару сокетов. Каждый процесс пользуется своим сокетом, при этом сокеты могу находиться как на разных компьютерах, так и на одном.
- Сокет может иметь как высокоуровневое символьное имя (адрес), так и низкоуровневое, отражающее специфику адресации определенного коммуникационного домена. Например, в домене Интернета низкоуровневое имя представлено парой (IP-адрес, порт).

АНАЛИЗ СТРУКТУРЫ ЛОКАЛЬНОЙ СЕТИ ФАКУЛЬТЕТА ПМИ

Структура локальной сети ФПМИ по состоянию на 01.09.2005, на 01.07.2009, на 10.10.2012 показана на рис. 1.2, 1.2*а*,1.2*б* соответственно.

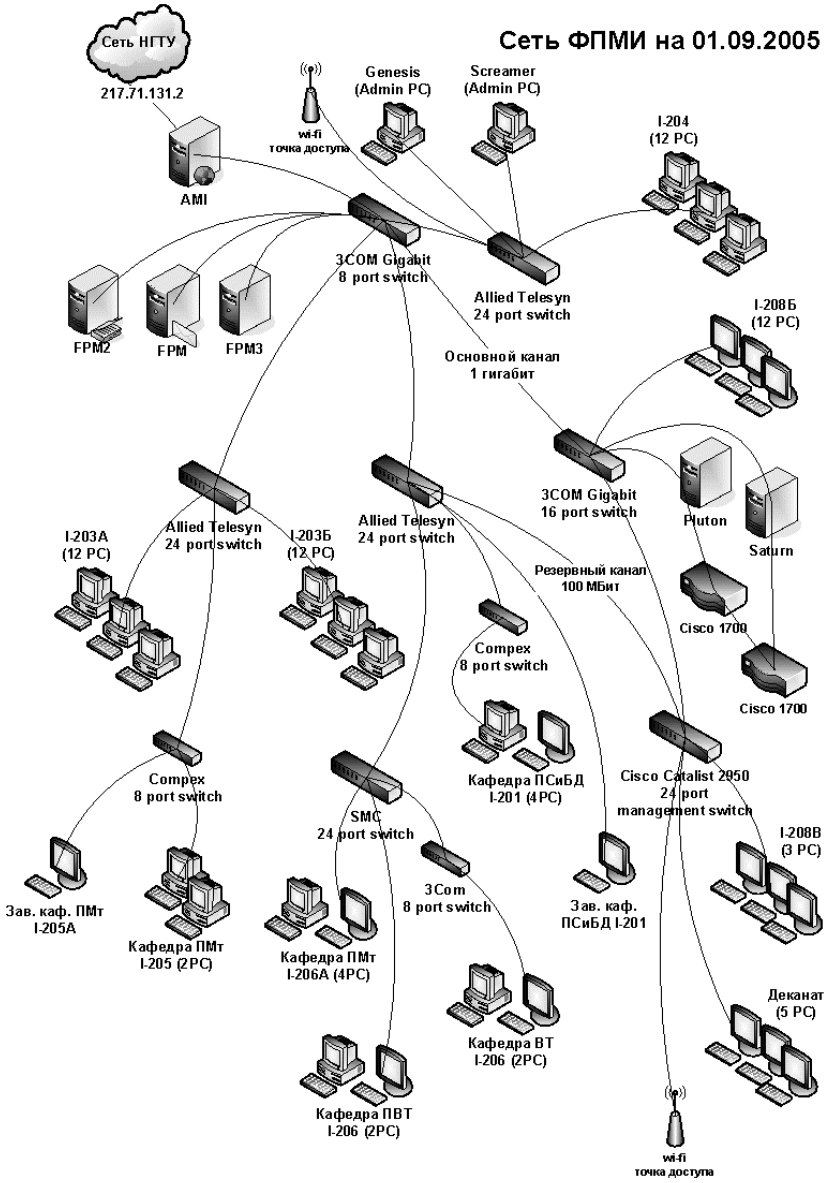


Рис. 1.2. Схема локальной сети факультета ПМИ

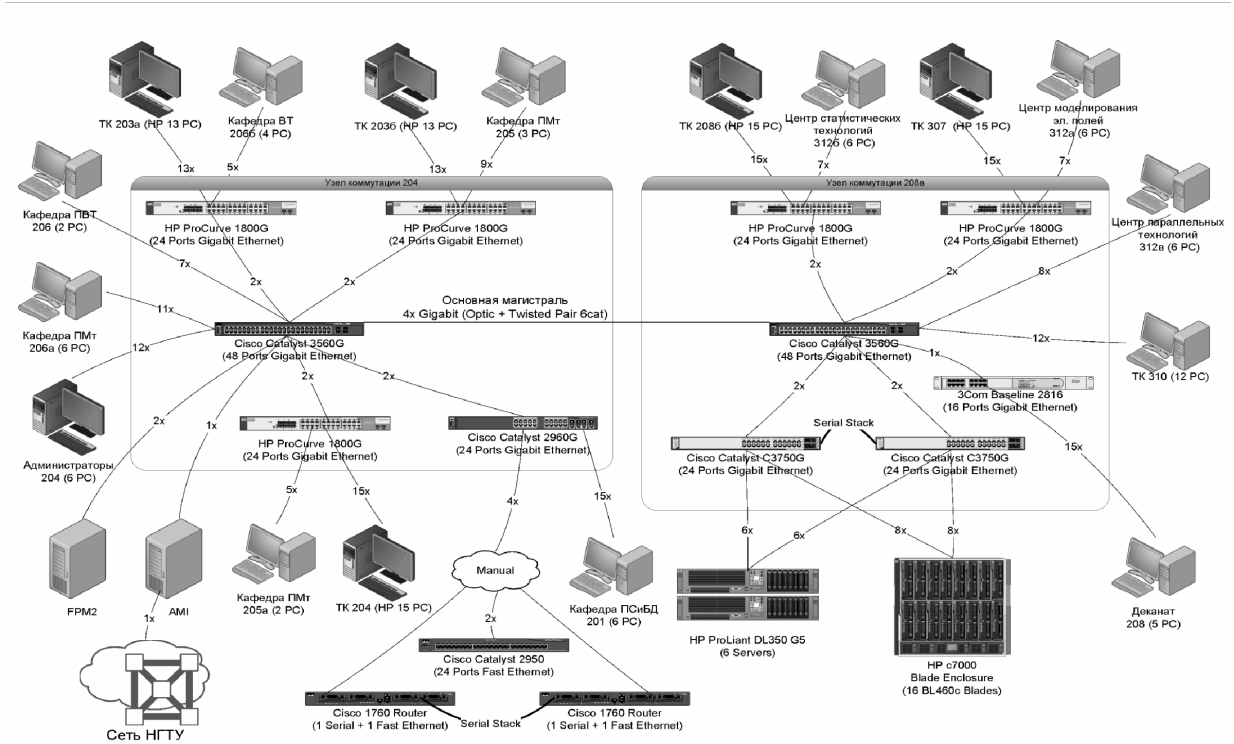


Рис. 1.2а. Сеть ФПМИ на 01.07.2009

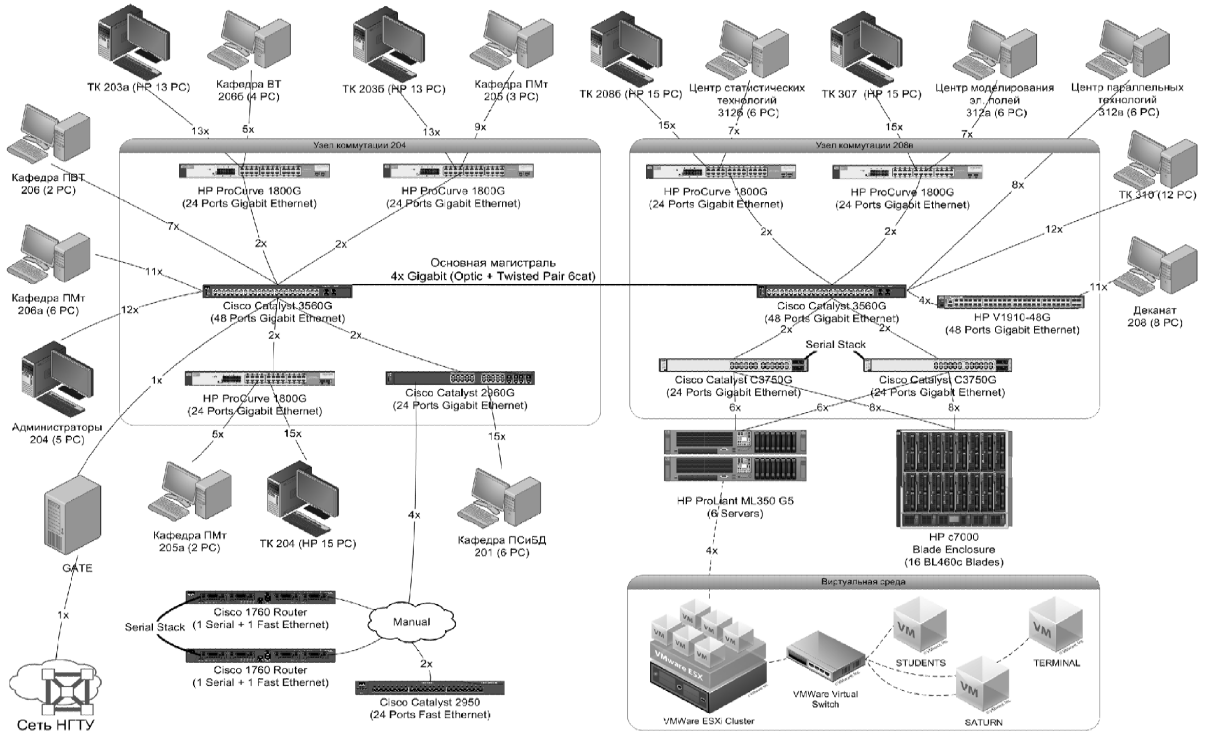


Рис. 1.2б. Сеть ФПМИ на 10.10.2012

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

1. Создать личную страничку (на бригаду) на сервере fpm2 для размещения отчетов о выполненных лабораторных работах. Рекомендуется личную страничку оформить в следующем виде.

Информация о бригаде (личная страница бригады №… группы ПМ…)		
Состав бригады	Учеба	Разное
Ф.И.О. Ф.И.О. ……	СИТ ПСЗИ …….	Новости ……
Версия № … от ДД.ММ. ГГ		

ЗАМЕЧАНИЕ 1

Один из простейших вариантов создания собственной страницы на WEB-сайте (Apache) для размещения отчетов по лабораторным работам:

A. Войти через PuTTY на сервер fpm2.ami.nstu.ru под бригадным логином и выполнить команду : **chmod -R 755 ~**

B. Создать в домашнем каталоге каталог с именем **public_html**, в этот каталог записать файл **index.html**, который должен содержать стартовый образ вашей страницы (см. п.1 задания на лабораторную работу)

C. Теперь содержимое файла **index.html** будет доступно по адресу следующего вида: http://fpm2.ami.nstu.ru/~ваш бригадный логин

Например, если ваша бригада – pm1403, то по адресу : http://fpm2.ami.nstu.ru/~pm1403 браузер откроет вашу начальную страницу **index.html** .

2. Выполнить анализ структуры локальной сети факультета по следующим пунктам:

- сетевые устройства, используемые в сети;
- линии связи, используемые в локальной сети факультета;
- схема соединения ПК*i* с сервером fpm2;
- структура сетевого программного обеспечения на каждом узле схемы соединения ПК*i* с сервером fpm2;

- IP и MAC-адреса ПК*i* и сервера fpm2.

3. Выполнить анализ директории CD в каталоге DNL.
4. Ответить на контрольные вопросы к лабораторной работе.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Дайте определение сетевому протоколу. Зачем он нужен?
2. Какой стек протоколов используется в сети Интернет?
3. В чем преимущества и недостатки многоуровневой организации системы?
4. Назовите виды единиц обмена информацией разных уровней в архитектуре протоколов.
5. Перечислите уровни стека протоколов Интернет и назовите их основные функции.
6. Перечислите известные вам виды сетевого оборудования и объясните, для чего они используются.
7. Перечислите виды физических сред передачи данных и их особенности.
8. В чем состоит принцип взаимодействия прикладной программы с системным программным обеспечением?
9. На каком уровне модели OSI работает коммутатор?
10. На каком уровне модели OSI работает шлюз?

Лабораторная работа № 2
ТЕХНОЛОГИЯ КЛИЕНТ-СЕРВЕР. ЭХО-ПОВТОР

Цель работы. Изучить основные принципы разработки клиент-серверных приложений на примере простейшей однопользовательской программы.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ И МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Модель, которая предусматривает, что прикладная программа должна пассивно ждать, пока другое приложение не инициирует связь, широко применяется в распределенных вычислениях и носит название «взаимодействие по принципу клиент/сервер».

Приложение, которое активно инициирует контакт, называется клиентом, а приложение, которое пассивно ожидает контакта, называется сервером.

Хотя и существуют небольшие различия, большинство реализаций средств взаимодействия клиент/сервер характеризуется общими особенностями.

Клиентская программа:

- представляет собой произвольную прикладную программу, которая становится клиентом на время, когда ей требуется удаленный доступ, но выполняет также другие локальные действия;
- вызывается непосредственно пользователем и действует на протяжении только одного сеанса;
- функционирует локально на персональном компьютере пользователя;
- активно инициирует контакт с сервером;

- может обращаться по мере необходимости к нескольким службам, но в определенный момент времени активно контактирует только с одним удаленным сервером;
- не требует специальных аппаратных средств или сложной операционной системы.
- Серверная программа:
- это программа специального назначения, которая выделена для предоставления одной службы, но может обслуживать нескольких удаленных клиентов одновременно;
- вызывается автоматически во время начальной загрузки системы и продолжает работать, проводя один сеанс взаимодействия за другим;
- выполняется на компьютере, предоставленном в общее пользование;
- пассивно ожидает поступления запросов на установление соединения от удаленных клиентов;
- принимает запросы от клиентов, но предоставляет единственную службу;
- требует применения мощных аппаратных средств и сложной операционной системы.
- В процессе обмена данными между большинством приложений Internet выполняется одна и та же последовательность операций:
- в начале запускается на выполнение серверное приложение и ожидает запроса на установление соединения от клиента;
- клиент обращается к серверу, указывая его местонахождение и передавая требование приступить к обмену данными;
- клиент и сервер обмениваются сообщениями;
- после завершения передачи данных и клиент, и сервер сообщают о том, что достигнут конец файла, чтобы прекратить обмен данными.

Общая схема взаимодействия клиента и сервера показана на рис. 2.1.

Взаимодействие клиента и сервера на транспортном уровне может происходить с использованием различных протоколов. Наиболее распространенными протоколами являются TCP и UDP. Приведем некоторые различия в схеме взаимодействия клиентской и серверной программ при использовании этих протоколов.

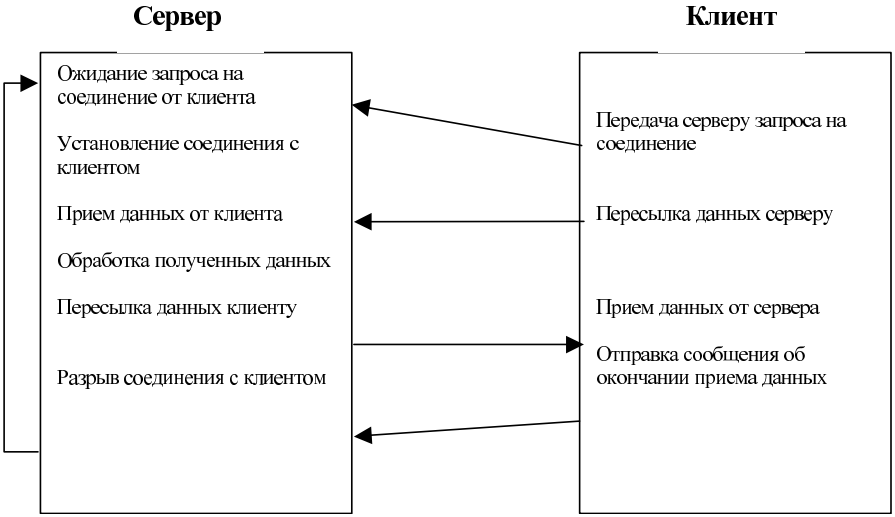


Рис. 2.1. Схема взаимодействия клиента с сервером

Протокол TCP поддерживает надежную передачу потока данных с предварительным установлением связи между источником информации и ее получателем. На базе протокола TCP реализованы такие протоколы уровня приложений, как Telneat, FTP, HTTP. Протокол характеризуется следующими особенностями:

- перед фактической передачей данных необходимо установить связь, т. е. запрос на начало сеанса передачи данных источником и подтверждение получателем;
- после обмена данными сеанс передачи должен быть явно завершен;
- доставка информации может быть надежной, не допускающей дублирования или нарушения очередности получения данных;
- существует возможность управления потоком данных для избежания переполнения и затора.

Эти возможности протокола позволяют протоколам верхнего уровня и приложениям, их реализующим, не заботиться о надежности и последовательности доставки данных. Поэтому протоколы приложений, использующих TCP, могут быть значительно упрощены. Но это, в свою очередь, ведет к сложности самого протокола, а значит, и значительным накладным расходам при передаче данных.

Протокол UDP обеспечивает логический коммуникационный канал между источником и получателем данных без предварительного установления связи. Для передачи дейтаграмм UDP используют протокол IP, который также не обеспечивает надежность передачи данных. Поэтому приложения, использующие данный протокол, должны самостоятельно отслеживать надежность доставки, например, путем обмена подтверждениями и повторной передачей недоставленных сообщений.

Протокол UDP используют такие протоколы уровня приложений, как протокол взаимодействия с сервером доменных имен DNS, протокол удаленного копирования Trivial FTP, удаленный вызов процедур RPC.

Благодаря минимальной функциональности протокола UDP передача данных с его использованием вносит гораздо меньшие накладные расходы по сравнению с протоколом TCP.

СХЕМА РАБОТЫ ЭХО-СЕРВЕРА И ЭХО-КЛИЕНТА

Сервер Запуск происходит с указанием номера порта из возможных номеров для протокола TCP/IP, например, 2000. Каждая бригада создает свой сервер с номером порта 2000 + номер бригады.

Сервер находится в режиме ожидания соединения от клиента. После установления соединения сервер переходит в бесконечный цикл по обработке данных, а именно приему данных от клиента и отправки их назад.

После отсоединения клиентского приложения сервер выходит из цикла и снова возвращается в режим ожидания соединения.

Клиент Запуск клиента происходит с указанием ему IP-адреса и TCP-порта сервера. Клиент устанавливает соединение с указанным сервером. После установления соединения клиентская программа выдает пользователю приглашение к вводу данных и входит в бесконечный цикл, в котором происходят чтение введенной строки и обработка данных с сервером, а именно отправка сообщений с буфера клавиатуры и отображение на экране полученных данных с сервера, после этого следует новое приглашение.

Остановка работы клиента должна происходить по любому заранее определенному ключевому слову, например exit.

Для реализации клиентского и серверного приложений можно использовать API-интерфейс сокетов низкого уровня. Ниже приведены список и подробное описание функций этого интерфейса.

ОПИСАНИЕ ФУНКЦИЙ РАБОТЫ С СОКЕТАМИ

Модуль socket предоставляет доступ из программы Python к сокетам. Сокетом называется конечная точка сетевых коммуникаций. Сокеты используют транспортный уровень согласно семиуровневой модели OSI (Open Systems Interconnection – взаимодействие открытых систем).

Для создания соединения TCP/IP необходимы два сокета: один на локальной машине, а другой на удаленной. Таким образом, каждое сетевое соединение характеризуется IP-адресом локальной машины и портом на локальной машине, IP-адресом и портом на удаленной машине.

При реализации модели «клиент-сервер» сервер обычно слушает (listen) порт с определенным номером. Дождавшись запроса от клиента на этот порт, сервер и клиент устанавливают соединение, используя свободные порты.

МЕТОДЫ, КОТОРЫЕ РЕАЛИЗУЮТ API-ИНТЕРФЕЙС СОКЕТОВ

Метод socket создает объект типа сокет и возвращает его:

socket.socket(af, type, [protocol]) -> sock_obj

Параметр af указывает, какое семейство протоколов будет использоваться с сокетом. Чаще всего используется значение AF_INET – домен Интернета. Параметр type указывает тип связи, который будет использоваться в сокете. Наиболее распространенными типами являются потоковая передача с установлением логического соединения (которая задается значением SOCK_STREAM) и блочная передача без установления логического соединения (которая задается значением SOCK_DGRAM). Дальнейшая работа с сокет-объектом проводится вызовом его методов.

Метод close сообщает операционной системе, что программа завершила использование сокета. Вызов имеет форму

close()

Сразу после создания сокет не имеет ни локального, ни удаленного адреса. В сервере для задания номера порта протокола, через который сервер будет принимать запросы на установление соединения, используется метод bind. Вызов метода bind имеет вид:

bind(address)

Вызов этой процедуры представляет собой запрос: назначить сокету конкретный номер порта протокола. Параметр address здесь и далее соответствует адресной информации, принятой в данной адресной схеме. Для сокетов, работающих на основе IP, адрес – пара (host, port), где host указывает IP-адрес хоста, а port – порт.

После задания номера порта протокола сервер должен передать операционной системе команду перевести сокет в пассивный режим, чтобы он мог применяться для получения запросов на установление соединения от клиентов. Для этого сервер вызывает метод listen, который принимает один параметр:

```
listen(queuesize)
```

Параметр queuesize указывает длину очереди запросов сокета.

Сервер, в котором используется транспортный протокол с установлением логического соединения, должен выполнить метод accept для приема следующего запроса на установление соединения. Если в очереди находится хотя бы один запрос, метод accept немедленно выполняет возврат; если же в очередь не поступило ни одного запроса, то система блокирует сервер до тех пор, пока один из клиентов не сформирует запрос на установление соединения. Метод accept имеет следующую форму:

```
accept() -> newsock, address
```

Процедура accept создает новый сокет-объект для соединения и возвращает его, а также адресную информацию о клиенте (address) вызывающей процедуре. Сервер использует новый сокет для обмена данными с клиентом и закрывает сокет после завершения работы. Первоначальный сокет сервера остается неизменным: после завершения обмена данными с клиентом сервер использует первоначальный сокет для приема следующего запроса на установление соединения от клиента.

В клиентских программах для установления соединения с конкретным сервером применяется метод connect. Вызов этого метода имеет форму:

```
connect(address)
```

Параметр address указывает адрес сервера.

И клиенты, и серверы должны передавать информацию. Обычно клиент посылает запрос, а сервер возвращает ответ. Если сокет под-

ключен, для передачи данных может применяться метод send. Метод send имеет два параметра, причем один из них необязательный:

```
send(data[, flags]) -> bytes
```

Параметр flags состоит из специальных констант, с помощью которых можно запрашивать специальные опции. Метод возвращает количество переданных байт.

Метод sendto позволяет клиенту или серверу передавать сообщение с использованием неподключенного сокета; он требует указывать адрес назначения. Он имеет следующую форму:

```
sendto(data[, flags], address) -> bytes
```

Эта процедура отличается от процедуры send только последним параметром, который задает адрес назначения.

И клиент, и сервер должны получать данные, переданные другим участником соединения. В API-интерфейсе сокетов предусмотрено несколько методов, которые могут использоваться для этой цели. Например, в приложении может быть выполнен метод recv для получения данных из подключенного сокета. Этот метод имеет следующую форму:

```
recv(buffsize[, flags]) -> data
```

Параметр buffsize определяет максимальный объем порции данных, получаемых за один вызов метода. Необязательный параметр flags аналогичен этому же параметру методов send и sendto. Метод возвращает данные, полученные из сокета.

Если сокет не подключен, его можно использовать для получения сообщений от произвольного набора клиентов. В таких случаях система возвращает адрес отправителя наряду с каждым входящим сообщением:

```
recvfrom(buffsize[, flags]) -> data, address
```

Параметры соответствуют параметрам процедуры recv. Кроме данных, полученных из сокета, метод recvfrom регистрирует адрес отправителя точно в такой же форме, в какой его принимает метод sendto.

Схема взаимодействия сервера с клиентом через интерфейс сокетов показана на рис. 2.2.

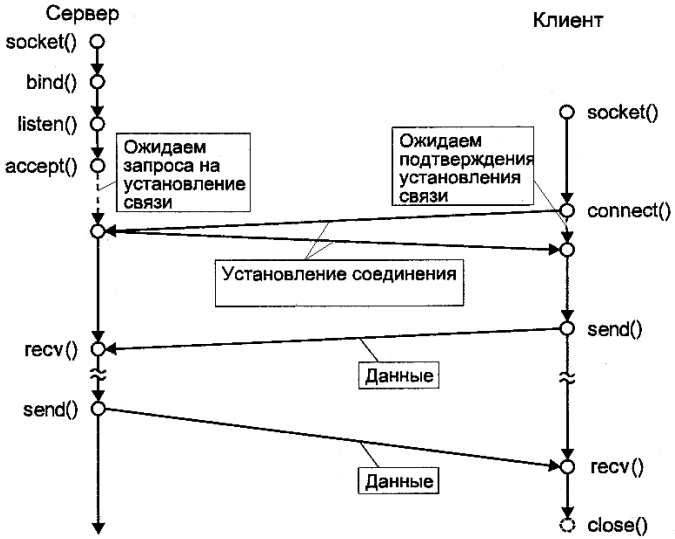


Рис. 2.2. Схема установления связи и передачи данных между клиентом и сервером

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

Написать простейшее приложение клиент-сервер с одним сервером и одним клиентом используя API-интерфейс низкого уровня. Пример интерфейса приведен в приложении.

Сервер начинает свою работу с ожидания запроса от клиента на соединение. Клиент устанавливает связь с сервером и посылает набор данных, введенный пользователем, на сервер. Сервер получает от клиента набор данных, выполняет указанные в варианте действия и возвращает клиенту результат. После этого сервер снова переходит в состояние ожидания запроса на соединение. Клиент, получив ответ с сервера, распечатывает его на экране и прекращает работу.

ВАРИАНТЫ ЗАДАНИЙ

1. Клиент пересылает серверу данные (строки текста). Сервер возвращает клиенту полученные данные, включив в конец каждого предложения количество символов в нем.

2. Клиент пересылает серверу данные (строки текста). Сервер изменяет порядок следования букв в полученном тексте на обратный и отправляет текст в таком виде клиенту.

3. Клиент пересылает серверу данные (строки текста). Сервер в полученном тексте в конец каждого предложения вставляет свой IP-адрес и номер порта и возвращает в таком виде данные клиенту.

4. Клиент пересылает серверу данные (строки текста). Сервер создает файл с уникальным именем, записывает в него полученные от клиента данные и в качестве результата обработки данных отправляет клиенту имя созданного файла. После получения ответа с сервера клиент распечатывает на экран содержимое указанного сервером файла.

5. Клиент пересылает серверу данные (строку и имя директории). Сервер находит все файлы в заданной директории, содержащие указанную строку, и высылает их имена клиенту.

6. Клиент пересылает серверу имя некоторого файла. Сервер находит файл с указанным именем и пересылает его содержимое клиенту либо сообщает клиенту, что файл с данным именем не найден.

7. Клиент пересылает серверу данные (имя директории). Сервер возвращает список файлов и поддиректорий данной директории (рекурсивно).

8. Клиент пересылает серверу два числа. Сервер возвращает сумму полученных чисел.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что представляет собой модель клиент-сервер?
2. Приведите общие особенности клиентских программ.
3. Приведите общие особенности серверных программ.
4. Приведите общую схему клиент-серверного взаимодействия.
5. В чем различие взаимодействия клиента с сервером при использовании различных протоколов транспортного уровня, таких как TCP и UDP? В чем преимущества и недостатки каждого из протоколов?
6. Что такое сокет? Какие виды сокетов вам известны?
7. Опишите основные методы работы с сокетами.
8. Приведите схему взаимодействия клиента с сервером при использовании механизма сокетов.

Лабораторная работа № 3

СОЗДАНИЕ ПРИЛОЖЕНИЯ ИНТЕРАКТИВНОЙ ПЕРЕПИСКИ

Цель работы. Изучить основные принципы разработки многопользовательских приложений, построенных на основе технологии клиент-сервер с использованием стека протоколов TCP/IP.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ И МЕТОДИЧЕСКИЕ УКАЗАНИЯ

В лабораторной работе № 2 было реализовано простейшее взаимодействие, в котором участвовали один клиент и один сервер. В данной лабораторной работе предлагается модифицировать программы таким образом, чтобы сервер мог осуществлять взаимодействие с несколькими клиентами сразу.

СХЕМА РАБОТЫ CHAT-СЕРВЕРА И CHAT-КЛИЕНТА

Сервер. Работа сервера начинается с перехода в состояние ожидания запроса на установление соединения от клиента. Затем сервер входит в цикл, в котором он получает и отображает строки текста от клиентов.

Клиент. Работа клиентской программы начинается с передачи серверу запроса на установление соединения. После установления соединения клиент также входит в цикл. При каждом проходе по циклу клиент выдает локальному пользователю приглашение к вводу строки текста, считывает строку, введенную с клавиатуры, отправляет ее на сервер, а затем получает и отображает строку текста, полученную с сервера (рис. 3.1).

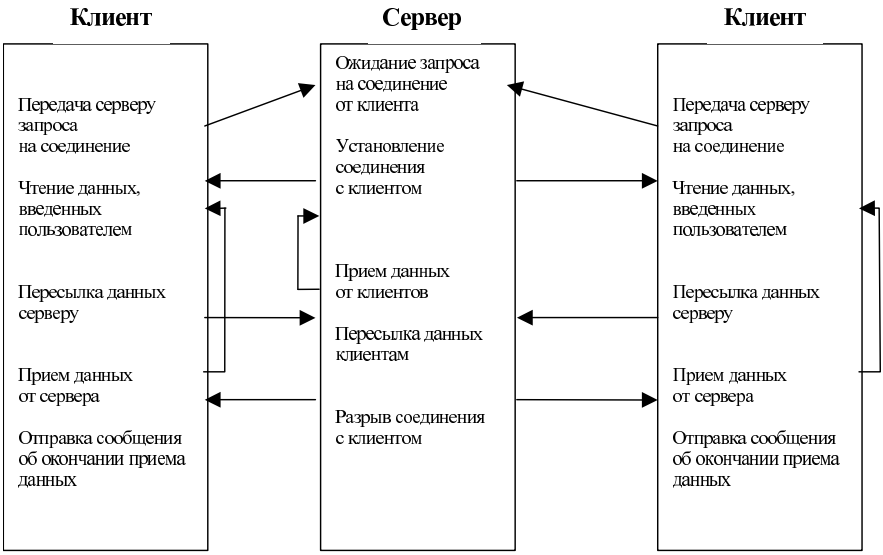


Рис. 3.1. Схема взаимодействия Chat-сервера с клиентами

ПОТОКИ УПРАВЛЕНИЯ

Поскольку сервер должен работать одновременно с каждым клиентом, рекомендуется либо использовать неблокирующие сокеты (читать в цикле данные из каждого сокета по очереди), либо для каждого клиента запускать отдельный параллельный поток (thread) операционной системы, который бы занимался обменом сообщениями с клиентом. Необходимые для работы с потоками функции содержатся в модулях thread и threading.

Потоки управления образуются и живут в рамках одного процесса. В однопоточном приложении есть только один поток управления: при запуске программы он последовательно исполняет встреченные в программе операторы. В любой момент времени интерпретатор знает, какую команду выполнить, а после выполнения команды – куда передать управление. Эта ниточка непрерывна все время исполнения программы и обрывается только по ее завершении.

Представим теперь, что в некоторой точке ниточка раздваивается и каждый поток пошел своим путем, возможно, еще несколько раз раз-

дваиваясь по пути. (При этом один из потоков всегда остается главным, и его завершение означает завершение всей программы.) В каждый момент времени интерпретатор знает, какую команду какой поток должен выполнить, и уделяет внимание каждому потоку. Однако такое незначительное усложнение выполнения программы приводит к тому, что нам необходимы механизмы для согласования деятельности потоков. Например, нельзя допускать, чтобы потоки одновременно изменяли один и тот же объект. На помощь приходят специальные объекты, называемые семафорами. Семафоры позволяют исключать выполнение одного и того же участка кода несколькими потоками одновременно. Самый простой двоичный семафор – замок (lock) или mutex. Чтобы поток мог продолжить выполнение кода, он должен сначала овладеть замком. После этого он выполняет некоторый участок кода и снимает замок, чтобы другой поток (возможно, уже сделавший запрос на данный замок) мог его получить и пройти дальше к выполнению охраняемого замком участка программы.

Класс **threading.Thread** позволяет назначить действия, которые должны выполняться в отдельном потоке, и имеет следующий конструктор:

threading.Thread(group,target,name,args=(),kwargs={}) -> object

где group – группа потоков; target – функция, метод или другой объект, позволяющий вызов; именно он вызывается при запуске потока, name – имя потока; args и kwargs – соответственно позиционные и именованные аргументы для вызова потока.

Приведем некоторые методы объектов класса **threading.Thread**. Метод **start()** запускает поток. (Метод не имеет параметров и ничего не возвращает.) Метод **run()** используется для вызова в отдельном потоке. Получает свое значение из параметра **target** конструктора класса. При вызове используются **args** и **kwargs**, заданные в конструкторе. Метод **join([time])** ожидает завершения потока. Поток, который вызывает этот метод, приостанавливается. Значение **time** задает время ожидания, после которого приостановленный поток продолжает свою работу. Метод **getName()** возвращает имя потока, метод **setName(name)** устанавливает имя потока. Метод **isAlive()** возвращает значение «истина», если поток работает.

Класс **threading.Lock** – простейший замок, который имеет два состояния (он может быть либо открыт, либо заперт). Объект **Lock** имеет два основных метода **acquire()** (с помощью этого метода поток делает

запрос на запирание замка) и **release()** (для снятия замка). Метод **locked()** возвращает статус замка: 0 – свободен, 1 – занят.

С помощью этого вида замка можно обеспечить выполнение некоторого участка программы одновременно только одним потоком. Еще раз запросить закрытый замок не может даже сам процесс, который его до этого запер: это приводит к бесконечному ожиданию. Однако отпереть замок может любой процесс.

В некоторых случаях (например, в рекурсивных функциях) необходимо, чтобы один и тот же поток имел возможность запрашивать замок даже в случае, если он им уже обладает. Объект класса **threading.RLock** имеет эту возможность и потому может сколько угодно раз запрашивать замок методом **acquire()** и столько же раз снимать методом **release().** Первоначально замок находится в открытом состоянии и не принадлежит ни одному потоку. Замок возвращается в исходное незапертое состояние только после одинакового числа запросов и освобождений. При этом все остальные потоки, которые запрашивают замок, остаются в состоянии ожидания.

Семафоры – класс **threading.Semaphore** – представляют собой более общий механизм синхронизации потоков, чем замки. С их помощью в критическую часть программы допускается несколько потоков. Семафор ведет счетчик запросов, который при каждом запросе **acquire()** уменьшается на единицу, а при каждом **release()** увеличивается на единицу. Счетчик не может стать меньше нуля, поэтому, если запрос поступает, когда счетчик равен нулю, потоку приходится ждать, как и в случае с замками, пока один из потоков не увеличит счетчик.

Объекты класса **threading.Event** служат для простейшей коммуникации между потоками, при которой один поток сигнализирует о событии, тогда как другие находятся в состоянии ожидания. Объекты события имеют внутренний флаг, который может быть установлен или сброшен. При своем создании флаг находится в сброшенном состоянии. Если флаг установлен, ожидания не происходит: соответствующий поток продолжает свою работу.

Модуль **thread** предоставляет низкоуровневый доступ к потокам управления. В этом модуле доступны следующие функции.

Создание нового потока для выполнения функции **function** с кортежем аргументов **args** и словарем **kwargs**. Поток завершается по выходу из функции **thread.start_new(func,args[,kwargs]).**

Функция **thread.exit()** возбуждает исключение **SystemExit,** которое, если оказалось неперехваченным, завершает выполняющийся поток.

Функция **get_ident() -> id** возвращает идентификатор потока, в котором выполняется эта функция.

Вызовом функции **thread.allocate_lock() -> lockobj** осуществляется создание объекта замка. На основе этого замка создан класс **threading.Lock**, поэтому методы для работы с этими объектами одинаковы.

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

С помощью API-интерфейса реализовать простой **chat.**

Каждая бригада должна написать **chat**-сервер и **chat**-клиента. Сервер должен поддерживать соединение сразу от нескольких клиентов. Обмен между клиентами осуществляется через сервер. При получении сообщения от какого-либо клиента сервер дублирует его на своем экране и оповещает всех подсоединенных клиентов, отправляя каждому из них данное сообщение. При подсоединении нового клиента к chat-серверу сервер оповещает каждого клиента о новом пользователе, посылая им его IP-адрес и имя.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Приведите схему взаимодействия **chat**-сервера и **chat**-клиента.
2. Что такое потоки управления, для чего они нужны и как они работают?
3. Какие проблемы возникают при использовании потоков управления и какие методы решения этих проблем существуют?
4. Опишите классы модуля threading.
5. Какие виды замков существуют? Расскажите о преимуществах и недостатках каждого из них.

Лабораторная работа № 4
СОЗДАНИЕ WEB-СЕРВЕРА

Цель работы. Изучить технологии создания Web-серверов, работающих на основе протокола HTTP, особенности архитектуры программного обеспечения Web-браузеров и основные типы Web-документов.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ И МЕТОДИЧЕСКИЕ УКАЗАНИЯ

ИНТЕРФЕЙС БРАУЗЕРА

World Wide Web (WWW) – крупномасштабный, оперативный репозитарий информации, в котором пользователи могут выполнять поиск с использованием интерактивной прикладной программы, называемой **браузером**. Большинство браузеров имеет интерфейс, организованный по принципу «указать и щелкнуть» – браузер отображает информацию на экране компьютера и позволяет перемещаться по разделам этой информации с помощью мыши. Отображаемая информация может включать и текст, и графику. Кроме того, часть информации, отображаемой на экране, выделена для указания того, что соответствующий элемент может быть выбран пользователем. При установке пользователем курсора над элементом, доступным для выбора, и нажатия кнопки мыши браузер отображает информацию, которая соответствует выбранному элементу.

Гипермедийный документ, доступный в Web, называется **страницей**; главная страница организации или отдельного лица называется **начальной страницей.** Поскольку страница может содержать много информационных элементов, ее формат должен быть выбран очень

тщательно, чтобы содержимое страницы мог интерпретировать любой браузер. В частности, браузер должен различать произвольный текст, графику и ссылки на другие страницы. Кроме того, в распоряжении автора страницы должны находиться средства описания общей компоновки документа (например, порядка, в котором должны быть представлены элементы).

Для каждой Web-страницы, содержащей гипермедийный документ, используется стандартное представление. Этот стандарт, известный под названием **языка гипертекстовой разметки** (HyperText Markup Language – HTML), позволяет автору страницы реализовать свои замыслы по отображению информации на странице и указать, что на ней находится.

ИДЕНТИФИКАЦИЯ СТРАНИЦЫ

Вызывая браузер на выполнение, пользователь должен указать начальную страницу, к которой он хочет обратиться. Указание конкретной страницы является сложной задачей по нескольким причинам. Во-первых, в сеть Web входит много компьютеров и страница может находиться на любом из них; во-вторых, каждый компьютер может содержать много страниц и каждой из них должно быть присвоено уникальное имя. В-третьих, Web обеспечивает отображение документов во многих форматах, поэтому браузеру должно быть дано указание, какое представление применяется на данной странице (например, относится ли указанное имя к документу HTML или к изображению, хранящемуся в двоичной форме). В-четвертых, поскольку сеть Web интегрирована с другими приложениями, браузер должен иметь информацию о том, какой прикладной протокол должен применяться для доступа к странице.

Разработан синтаксический формат, который позволяет обозначить элемент данных, находящийся на удаленном компьютере. Этот синтаксический формат позволяет закодировать всю информацию в символьной строке, называемой **унифицированным локатором ресурсов** (Uniform Resourse Locator-URL). URL имеет следующую общую форму:

protocol://computer_name:port/document_name

Здесь protocol – имя протокола, применяемого для доступа к документу; computer_name – доменное имя компьютера, на котором находится

документ; :port – необязательный номер порта протокола; document_name – имя документа, под которым он хранится на указанном компьютере.

ВЗАИМОДЕЙСТВИЕ ТИПА КЛИЕНТ/СЕРВЕР

Как и в других сетевых приложениях, в программах просмотра Web применяется модель взаимодействия типа клиент/сервер. После получения URL документа браузер становится клиентом, который обращается с запросом на получение документа к серверу, работающему на компьютере, указанном в URL. Затем браузер отображает документ для пользователя.

В отличие от сетевых приложений, в которых предусмотрено постоянное соединение между клиентом и сервером, при взаимодействии между Web-браузером и сервером соединение создается лишь на короткое время. Браузер устанавливает соединение, отправляет запрос и получает либо затребованный элемент данных, либо сообщение, что такого элемента не существует. Сразу после передачи документа или изображения соединение закрывается; клиент не остается подключенным к серверу.

Быстрое прекращение соединений вполне себя оправдывает в большинстве случаев, поскольку просмотр Web не характеризуется высокой степенью локализации связей. Пользователь может обратиться к Web-странице на одном компьютере, а затем немедленно последовать по ссылке к Web-странице на другом компьютере. Однако слишком быстрое прекращение соединения может привести к ненужным издержкам в тех случаях, если браузер должен неоднократно обращаться к одному и тому же серверу для получения большого числа документов.

ПЕРЕДАЧА ДОКУМЕНТОВ WEB И ПРОТОКОЛ HTTP

При взаимодействии браузера с Web-сервером эти две программы выполняют протокол HTTP (HyperText Transfer Protocol – протокол передачи гипертекста). Назначение протокола HTTP состоит в том, что он позволяет браузеру запросить конкретный элемент данных, который затем отправляет ему сервер. Однако на практике с применением протокола HTTP связано много сложностей, поскольку сервер вместе с каждой передачей отправляет дополнительную информацию состоя-

ния, а браузер, согласно этому протоколу, может не только запрашивать, но и передавать информацию.

Запросы HTTP передаются в виде текста в кодировке ASCII. Протокол HTTP поддерживает четыре основные операции, которые могут быть указаны браузером при выполнении запроса.

- **Операция GET** позволяет запросить конкретный элемент данных с сервера. Сервер возвращает заголовок, за которым следуют информация состояния, пустая строка и затребованный элемент данных.
- **Операция HEAD** позволяет запросить информацию о состоянии некоторого элемента. Сервер возвращает информацию состояния, не передавая копию самого элемента.
- **Операция POST** служит для передачи данных на сервер. Сервер добавляет переданные данные к указанному элементу (например, добавляет сообщение к списку сообщений).
- **Операция PUT** также позволяет передать данные на сервер. Однако сервер использует эти данные для замены указанного элемента.

При вводе пользователем URL или выборе ссылки браузер формирует запрос HTTP. В том или ином случае браузер передает запрос GET, в котором указан определенный элемент данных, и сервер возвращает запрашиваемый элемент. Запрос GET имеет следующую форму:

GET item version CRLF

Здесь item обозначает URL затребованного элемента, **version** указывает версию HTTP (обычно 1.0 или 1.1), а CRLF обозначает символы CR (сокращение от **carriage return** – возврат каретки) и LF (сокращение от **linefeed** – перевод строки) кодировки ASCII.

Каждый ответ от сервера начинается с заголовка в кодировке ASCII. Первая строка заголовка содержит код состояния, который сообщает браузеру, успешно ли обработан сервером запрос. Если запрос был сформирован неправильно или затребованный в нем элемент недоступен, то код состояния позволяет выявить эту проблему. Например, сервер возвращает широко известный код состояния 404, если затребованный элемент не удалось найти. Успешно выполнив запрос, сервер возвращает код состояния 200; дополнительные строки заголовка предоставляют более полную информацию об элементе данных, такую как его длина, время последнего изменения и тип информационного наполнения. Пример заголовка HTTP приведен ниже.

HTTP/1.0 200 OK
Date: Mon, 30 Oct 2000 01:22:22 GMT
Server: Apache/1.2.5
Last-Modified: Sat, 28 Oct 2000 01:03:37 GMT
ETag: «130fe-81-3883bbe9»
Content-Length: .129
Accept-Ranges: bytes
Connection: close,
Content-Type: text/plain

Код состояния 200 в первой строке указывает, что сервер успешно выполнил запрос; остальные строки содержат дополнительную информацию о затребованном элементе данных.

АРХИТЕКТУРА ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ БРАУЗЕРА

Программы Web-браузеров имеют более сложную структуру по сравнению с Web-серверами. Сервер постоянно выполняет одну и ту же несложную задачу: ожидает, пока какой-либо браузер не откроет соединение и не затребует страницу. Затем сервер передает копию затребованного элемента данных, закрывает соединение и ожидает следующего запроса на установление соединения. Основную часть операции по обеспечению доступа к документу и его отображению выполняет браузер. В связи с этим браузер содержит несколько крупных программных компонентов, которые тесно взаимодействуют друг с другом, создавая впечатление бесперебойно функционирующей службы. В частности, браузер включает целый ряд клиентов, широкий перечень интерпретаторов, а также контроллер, который управляет этими программными компонентами. Контроллер образует центральную часть браузера. Он интерпретирует и щелчки мышью, и ввод с клавиатуры, а также вызывает другие компоненты для выполнения операций, указанных пользователем. Например, при вводе пользователем URL или щелчке на гипертекстовой ссылке контроллер вызывает клиентскую программу для выборки затребованного документа с удаленного сервера, на котором он находится, а затем вызывает интерпретатор для отображения документа на экране пользователя.

Браузер должен содержать интерпретатор HTML для отображения документов. Другие интерпретаторы являются необязательными. На вход интерпретатора HTML поступает документ, который соответствует синтаксическим правилам HTML; на выход интерпретатора

поступает отформатированная версия документа, предназначенная для отображения на экране пользователя. Интерпретатор выполняет компоновку, преобразуя спецификацию HTML в команды, подходящие для аппаратного обеспечения дисплея, за которым работает пользователь. Например, встретив в документе тег заголовка раздела, интерпретатор изменяет размер текста, который служит для изображения заголовка раздела. Встретив тег разрыва строки, интерпретатор открывает новую строку вывода.

Одна из наиболее важных функций интерпретатора HTML относится к выбираемым элементам. Интерпретатор должен хранить информацию о том, как связаны между собой позиции на экране и элементы документа HTML, относящиеся к каждому анкеру. При выборе пользователем любого элемента с помощью мыши браузер сравнивает информацию о текущей позиции курсора и хранимую информацию о положении анкеров, определяя, был ли пользователем выбран анкер.

Кроме клиента HTTP и интерпретатора HTML, браузер может включать компоненты, позволяющие выполнять дополнительные задачи. Например, многие браузеры включают клиент FTP, который применяется для доступа к службе передачи файлов. Некоторые браузеры содержат также программное обеспечение клиента электронной почты, позволяющее передавать и принимать сообщения по электронной почте.

КЭШИРОВАНИЕ В WEB-БРАУЗЕРАХ

Взаимодействие между клиентом и сервером при просмотре Web отличается от взаимодействия во многих клиент-серверных приложениях. На это есть две причины. Во-первых, поскольку пользователи обычно просматривают Web-страницы, находящиеся за пределами их организации, Web-браузеры обращаются к страницам на удаленных компьютерах чаще, чем к локальным страницам. Во-вторых, поскольку пользователи не выполняют повторный поиск одной и той же информации, они, как правило, не повторяют доступ к одним и тем же ресурсам.

Таким образом, просмотр Web-страницы имеет иной характер локализации связей по сравнению с другими приложениями, поэтому для оптимизации производительности браузеров используются не такие методы, как в других приложениях. В частности, ни браузеры, ни Web-

серверы не оптимизированы с учетом физической локализации связей. Кроме того, в браузерах применяются совершенно иные методы оптимизации с учетом временной локализации связей.

Как и в других приложениях, в браузерах для ускорения доступа к документу применяется кэш. Браузер помещает копию каждого полученного элемента данных в кэш на локальный диск. При выборе пользователем элемента, прежде чем обратиться за свежей копией этого элемента, браузер проверяет кэш на диске. Если кэш содержит нужный элемент, браузер получает его копию из кэша и обращается к серверу-источнику (т. е. к серверу, на котором находится просматриваемая страница), если элемент данных, принадлежащий к странице, не может быть найден в кэше.

Хранение элементов данных в кэше позволяет резко повысить производительность, поскольку браузер может считывать их с диска, не ожидая передачи этих элементов по сети. Кэширование особенно важно при просмотре больших страниц или при использовании низкоскоростных сетевых соединений.

Несмотря на значительное повышение скорости, хранение элементов данных в кэше в течение продолжительного времени не всегда оправданно. Во-первых, кэш может занимать огромные объемы дискового пространства. Во-вторых, повышение производительности происходит, только если пользователь снова возвращается к просмотру того же элемента данных. К сожалению, пользователи часто просматривают ресурсы Web только в поисках конкретной информации; найдя необходимую информацию, пользователи прекращают просмотр. В подобных ситуациях кэширование снижает производительность, поскольку браузер расходует дополнительное время для записи на диск ненужной информации.

Для того чтобы пользователи могли управлять использованием браузера кэша, в большинстве браузеров предусмотрены средства, которые дают возможность пользователю изменять правила работы с кэшем. Пользователь может установить лимит времени для кэширования, и браузер будет удалять элементы данных из кэша по истечении установленного времени. Браузеры обычно сопровождают кэш на протяжении конкретного сеанса. Пользователь, который не желает оставлять элементы в кэше от одного сеанса к другому, может установить лимит времени хранения элементов данных в кэше равным нулю. В подобных случаях браузер очищает весь кэш после завершения сеанса работы пользователя.

ПОДДЕРЖКА КЭШИРОВАНИЯ ПРОТОКОЛОМ HTTP

В протоколе HTTP предусмотрены определенные средства, которые обеспечивают кэширование. При формировании заголовка HTTP сервером может быть указан тайм-аут кэша для страницы. Если страница содержит информацию, которая постоянно изменяется, при формировании ее сервером может быть указано, что страница не предназначена для кэширования. Кроме того, браузер может отправить запрос с HTTP-заголовком, где указано, что для ответа на этот запрос не должны использоваться данные из кэша.

Операция HEAD протокола HTTP также позволяет управлять кэшем, поскольку значения тайм-аута кэша обычно всегда выбираются с учетом возможности развития событий по худшему сценарию (т. е. сервер выбирает небольшое значение тайм-аута, если есть хоть малейшее подозрение, что документ может измениться). Чтобы понять, как может операция HEAD применяться для оптимизации доступа к документу, рассмотрим браузер, в кэш которого записан большой документ. Предположим, что пользователь запросил этот документ после истечения тайм-аута кэша. Вместо загрузки свежей копии браузер может предпринять попытку снова обратиться к его кэшированной версии. Для этого браузер передает запрос с операцией HEAD к этому документу. Если информация состояния, возвращенная сервером, подтвердит, что кэшированная копия все еще действительна, то браузер отображает эту копию для пользователя. В ином случае браузер передает запрос GET для получения новой копии.

АЛЬТЕРНАТИВНЫЕ ПРОТОКОЛЫ ПЕРЕДАЧИ

Некоторыми группами разработчиков были сделаны попытки расширить HTTP и применить альтернативные протоколы. В версии 1.1 протокола HTTP реализован принцип устойчивых соединений, согласно которому одно соединение TCP используется для нескольких операций передачи (т. е. для выполнения нескольких запросов GET, поступающих от браузера). В протоколе HTTP-NG (сокращение от **Next-Generation of HTTP** – протокол HTTP следующего поколения) применяется двоичная кодировка заголовков вместо кодировки ASCII. Протокол HTTPS предусматривает защищенную передачу информации по протоколу HTTP (в нем для обеспечения конфиденциальности данных, передаваемых в сеансе HTTP, применяется шифрование).

К числу альтернативных протоколов, получивших наиболее широкое распространение, относится WAP (Wireless Access Protocol – протокол беспроводного доступа). Набор протоколов WAP, разработанный сообществом поставщиков, который известен под названием **WAP Forum**, предназначен для использования в портативных устройствах, таких как сотовые телефоны, пейджеры двусторонней связи и карманные органайзеры. Протокол WAP не просто заменяет HTTP, а включает широкий набор протоколов, предназначенных для решения конкретных проблем. Например, спецификация WAP определяет прикладную среду, стандарты разметки страниц и представления данных, сетевые транспортные протоколы, протокол защиты, протокол речевого телефонного интерфейса, протокол транзакций и язык сценариев. Короче говоря, спецификация WAP не опирается на существующие стандарты, а определяет полностью новый стек протоколов.

Безусловно, проектировщики WAP стремятся обеспечить взаимодействие устройств WAP с существующими Web-серверами. Поэтому в спецификации WAP предусмотрены proxy-серверы, которые обеспечивают такое взаимодействие. При запросе Web-страницы с устройства WAP proxy-сервер получает эту страницу с Web-сервера и перенаправляет ее на устройство, с которого поступил запрос, используя WAP. Если Web-страница представлена на языке HTML, proxy-сервер вызывает фильтр HTML для перевода страницы с языка HTML на язык разметки WAP.

Итак, World Wide Web представляет собой распределенное хранилище гипермедийной информации, доступ к которой осуществляется с помощью интерактивного браузера. Основная часть документов Web представлена на языке HTML, несмотря на то что было предложено несколько альтернативных языков. Кроме текста, документ на языке HTML содержит теги, которые определяют компоновку документа и форматирование его элементов. Изображения не включаются непосредственно в документ – тег в документе указывает место, куда должно быть вставлено изображение, и источник изображения. Для указания элементов документа HTML, которые ссылаются на внешние ресурсы, применяется тег анкера. При отображении документа браузер отмечает элементы, предназначенные для выбора; если пользователь выбирает эти элементы, браузер следует по ссылке на внешний ресурс для получения нового документа.

Ссылки на внешние ресурсы представлены в форме унифицированного локатора ресурсов (URL). Браузер извлекает из URL такую информацию, как имя протокола, применяемого для доступа к элементу данных, имя компьютера, на котором находится элемент, и имя самого элемента данных.

Программа браузера состоит из контроллера, одного или нескольких клиентов, используемых для доступа к документам, и одного или нескольких интерпретаторов для отображения документов. С целью повышения эффективности выборки документов в Web-браузере применяется кэширование. Копия каждого документа или изображения, просматриваемого пользователем, помещается в кэш; вместо получения новой копии с сервера-источника для следующих запросов применяется кэшированная копия. Протокол HTTP, используемый для передачи документов Web, предусматривает применение заголовков, в которых описан документ и указано, как долго должен документ храниться в кэше.

ОСНОВНЫЕ ТИПЫ ДОКУМЕНТОВ WEB

В целом документы Web могут быть разбиты на три категории.

- *Статический.* Статический документ Web находится в файле, который связан с Web-сервером. Автор статического документа определяет его содержимое при составлении документа. Поскольку содержимое документа не меняется, каждый запрос к этому документу приводит к получению одного и того же ответа.
- *Динамический.* Динамический документ Web не существует в окончательной форме. Он создается Web-сервером, когда браузер запрашивает этот документ. При поступлении запроса Web-сервер вызывает на выполнение прикладную программу, которая создает динамический документ. Сервер возвращает вывод программы в качестве ответа браузеру, с которого поступил запрос. Поскольку в ответ на каждый запрос документ всегда создается заново, содержимое динамического документа может изменяться.
- *Активный.* Активный документ не определяется сервером полностью. Он содержит компьютерную программу, в которой реализованы алгоритмы вычисления и отображения результатов. Когда браузер запрашивает активный документ, сервер возвращает копию программы, которая должна быть выполнена браузером на локальном компьютере. Программа активного документа при выполнении может взаимо-

действовать с пользователями, непрерывно изменять отображение. Поэтому содержимое активного документа никогда не бывает постоянным: документ изменяется до тех пор, пока пользователь не остановит выполнение программы.

ПРЕИМУЩЕСТВА И НЕДОСТАТКИ ДОКУМЕНТОВ КАЖДОГО ТИПА

Основные преимущества статического документа – это простота, надежность и скорость воспроизведения. Поскольку статический документ непосредственно содержит простые спецификации форматирования, он может быть создан любым лицом, даже не имеющим подготовки в области компьютерного программирования. Созданный статический документ всегда останется действительным. И, наконец, браузер может быстро отобразить статический документ, затем поместить его копию в кэше на локальном диске для ускоренного выполнения последующих запросов на получение этого документа.

Основной недостаток статического документа заключается в отсутствии гибкости: документ должен пересматриваться при каждом изменении содержащейся в нем информации. Кроме того, изменения всегда трудоемки, поскольку редактирование файла требует участия человека. Поэтому в статическом документе не может быть представлена информация, которая часто меняется.

Основное преимущество динамического документа – его способность представлять актуальную информацию. Например, динамический документ может применяться для получения информации о текущих ценах на акции, сообщений о погоде или о наличии билетов на концерт. При запросе информации браузером сервер вызывает на выполнение приложение, которое получает доступ к нужной информации и создает документ. Затем сервер передает документ на браузер.

При создании динамических документов сервер выполняет дополнительные функции, а в браузере для получения динамической страницы используются такие же способы взаимодействия с сервером, как и при получении статической страницы. С точки зрения браузера динамические страницы невозможно отличить от статических страниц. Поскольку и в статических, и в динамических страницах используется язык HTML, браузер не имеет информации о том, была ли страница извлечена сервером из файла на диске или получена динамически из компьютерной программы.

Основные недостатки подхода с использованием динамических документов связаны с увеличением затрат на создание документа и с неспособностью динамических документов отображать изменяющуюся информацию. Полученная браузером копия динамического документа не меняется. Поэтому информация динамического документа начинает устаревать сразу после его передачи браузеру. Предположим, что динамический документ содержит текущий биржевой отчет о ценах на акции. Поскольку цены на акции быстро меняются, этот документ может устареть еще до того, как пользователь закончит просматривать его содержимое.

Создание динамического документа и доступ к нему требуют больше издержек по сравнению со статическим документом. Расходы на оплату труда разработчиков, создающих динамические документы, выше по сравнению с оплатой за документы, поскольку для создания динамической Web-страницы необходимо писать компьютерную программу. Кроме того, эта программа должна быть тщательно спроектирована и всесторонне проверена, чтобы ее вывод всегда был действительным. Проверить правильность такой программы сложно, так как на вход могут поступать разнообразные значения данных, полученные из нескольких источников.

Кроме повышения расходов на создание динамических документов, увеличивается стоимость аппаратного обеспечения, поскольку для эксплуатации такого сервера должна применяться более мощная компьютерная система. И наконец, дли выборки динамического документа требуется немного больше времени по сравнению со статическим документом, так как сервер вынужден затрачивать дополнительное время на выполнение прикладной программы, предназначенной для создания документа.

Хотя динамический документ создается при получении запроса, информация в нем может быстро стать неактуальной. Основное преимущество активного документа перед динамическим документом заключается в его способности непрерывно обновлять информацию. Например, только активный документ может изменять отображение на экране достаточно быстро, для того чтобы на нем появилось анимационное изображение. Кроме того, активный документ может обращаться непосредственно к источникам информации и непрерывно обновлять отображение. Например, активный документ, который показывает цены на акции, может осуществлять выборку биржевой информацию и

менять отображение, не требуя каких-либо действий со стороны пользователя.

Основные недостатки активных документов заключаются в дополнительных затратах на создание и выполнение таких документов, а также в отсутствии защиты. Во-первых, для отображения активного документа необходимо более сложное программное обеспечение браузера, и сам браузер должен работать на мощной компьютерной системе. Во-вторых, разработка успешно функционирующих активных документов требует более квалифицированного программирования по сравнению с другими формами, а проверка созданных документов более сложная. В частности, поскольку активный документ должен работать на любом клиентском компьютере, а не на конкретном сервере, программа должна быть написана так, чтобы была исключена зависимость от средств, доступных в одной компьютерной системе, но не доступных в другой, И наконец, активный документ создает потенциальный риск нарушения защиты, поскольку он может передавать информацию не только с сервера на компьютер пользователя, но и с компьютера на сервер.

СХЕМА РАБОТЫ WEB-СЕРВЕРА ЛОГИКА РАБОТЫ WEB-СЕРВЕРА ПОД WINDOWS И LINUX В ЦЕЛОМ ОДИНАКОВА

Самое сложное в Web-сервере – это работа с сокетами. Схема работы при этом выглядит примерно так:

- инициализируем сокеты;
- создаем socket – ОС выдаст нам новый сокет;
- настраиваем сокет (setsockopt);
- ассоциируем (bind) этот сокет с IP-адресом и портом;
- начинаем «слушать» (listen) сокет – вообще, слушает ОС, а мы только обрабатываем данные;
- в цикле принимаем (accept) соединения – accept возвращает управление и новый сокет, как только появится запрос на указанные ранее порт и IP-адрес (иначе вернет INVALID_SOCKET);
- начинаем асинхронный (для проверки/тестированно все можно и в одном потоке делать) прием (recv) из полученного сокета;
- обрабатываем принятые данные;
- отправляем ответ (send);

- после окончания приема закрываем сокет;
- возвращаемся в начало цикла.

Примечания. При асинхронной обработке цикл будет только получать новые сокеты и отдавать их рабочим потокам.

В скобках – вызовы АПИ-сокетов, под линуксом почти все то же самое.

Под Windows все сокеты в Ws2_32.dll/

Подробности в allasm.ru/set_12.php.

Обращение к webserver будет происходить по следующему шаблону URL:

http://имя_компьютера:порт, например, http://pc-204-10:20001

Web-Server должен работать по протоколу HTTP версии 1.0 или 1.1.

Так как протокол HTTP – это надстройка над стеком протоколов TCP/IP, в котором связь между клиентом и сервером осуществляется точно так же, как и при работе с сокетами; сообщения, передаваемые на сервер и обратно клиенту, должны иметь определенный формат.

Клиент, делающий запрос, передает на сервер следующие строки:

```
GET / HTTP/1.1
Accept: */*
Accept-Language: ru
Accept-Encoding: gzip, deflate X4
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)
Host: ninja:3000
Connection: Keep-Alive
(compatible; MSIE 6.0; Windows NT 5.1)
```

значащими строками здесь являются GET / HTTP/1.1

GET – команда запроса у сервера какой-либо информации

/ – путь к файлу (По умолчанию корневой)

HTTP/1.1 – версия протокола

Сервер, в случае успешного ответа, посылает :

```
HTTP/1.0 200 OK                 // протокол, результат, код ошибки
Server: CNAI Demo Web Server    // имя web сервера
Content-Length: 235             // длина html файла
Content-Type: text/html         // тип файла – текстовый html
```

Об остальных типах сообщений можно почитать в Интернете, сделав соответствующий поиск по протоколу HTTP 1.1, либо в rfc документах, например RFC**2616**.

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

Каждая бригада должна написать свой собственный Web-сервер, который должен отображать их собственную страницу. В качестве клиентов будет использоваться любой browser, например, Internet Explorer, Mozilla и т. п.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Интерфейс Web-браузера.
2. Идентификация Web-страниц.
3. Что такое унифицированный локатор ресурсов? Приведите пример.
4. Как происходит взаимодействие Web-браузера и сервера?
5. В чем назначение протокола HTTP? Приведите его основные операции.
6. Как формируется запрос Web-браузера?
7. Приведите пример заголовка HTTP, возвращаемого сервером.
8. Структура Web-браузера.
9. Кэширование в Web-браузерах.
10. Поддержка кэширования протоколом HTTP.
11. Альтернативные протоколы передачи.
12. Основные типы документов Web.
13. Преимущества и недостатки документов каждого типа.

Лабораторная работа № 5

АНАЛИЗ СТРУКТУРЫ КАДРА/ФРЕЙМА ТЕХНОЛОГИИ ETHERNET

Цель работы. Спроектировать и реализовать программу, выполняющую анализ структуры кадра/фрейма технологии Ethernet.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ И МЕТОДИЧЕСКИЕ УКАЗАНИЯ

ПЕРЕДАЧА ФРЕЙМОВ ПО СЕТИ

При передаче фрейма по сети Ethernet электрические сигналы, несущие биты, достигают всех станций сегмента локальной сети. Сетевые интерфейсные аппаратные средства этих станций обнаруживают электрический сигнал и могут принять копию фрейма.

В локальных сетях разного типа для обеспечения непосредственной связи применяется та или иная **схема адресации**. Каждой станции локальной сети присваивается уникальное числовое значение, называемое *физическим адресом, аппаратным адресом*, или *адресом доступа к передающей среде* (MAC-Media Access Control). При передаче фрейма по локальной сети отправитель включает во фрейм аппаратный адрес намеченного получателя. Аппаратное обеспечение каждой станции проверяет адрес каждого входящего фрейма, чтобы определить, должна ли станция принять этот фрейм.

Фрейм, передаваемый по разделяемой локальной сети, включает два адреса; один из них указывает получателя, а другой – отправителя. Каждый фрейм начинается с заголовка фиксированного формата, в котором отведено место для двух адресов. Участки заголовка, зарезерви-

рованные для адресов, принято называть **полями.** При передаче фрейма отправитель должен поместить физический адрес получателя в поле адреса назначения, а свой адрес – в поле адреса источника. Благодаря наличию адреса отправителя в каждом фрейме упрощается задача подготовки ответа получателем. Сетевые интерфейсные аппаратные средства обеспечивают проверку полей адресов в проходящих по сети фреймах и прием только тех фреймов, в которых адрес назначения совпадает с адресом станции.

СПОСОБЫ АДРЕСАЦИИ

Способы адресации можно разбить на следующие категории:

- статическая адресация;
- настраиваемая адресация;
- динамическая адресация.

В статической схеме физические адреса назначаются изготовителем аппаратных средств. Статический физический адрес станции не меняется до тех пор, пока на ней не произойдет замена аппаратных средств.

Схема **настраиваемой адресации** позволяет заказчику устанавливать физический адрес. Этот механизм может предусматривать ручную настройку (т. е. перестановку переключателей при инсталляции интерфейса) или электронную настройку. Большинство аппаратных средств требует настройки конфигурации только один раз; настройка обычно выполняется при первоначальной установке аппаратных средств.

Схема **динамической адресации** позволяет автоматически присваивать станции физический адрес сразу после ее загрузки. Большинство схем динамической адресации требует, чтобы станция случайным образом проверяла номера до тех пор, пока не будет найдено значение, не используемое другими компьютерами в качестве адреса. Например, станцией может быть выбрано в качестве начального значения адреса текущее время суток. После генерации случайного числа станция посылает по сети сообщение с соответствующим адресом. Если этот адрес уже используется каким-либо компьютером, то он отвечает на это сообщение. Если же никакая станция не ответит на отправленный запрос, отправитель может использовать адрес в качестве своего физического адреса. Таким образом, адрес, выбранный компьютером, зависит от того, какие адреса применяются другими компьютерами ко времени

его загрузки: при каждом последующем перезапуске компьютер может получать другой адрес.

К основным **преимуществам статической адресации** относятся удобство использования и постоянство. Эту схему легко применять, поскольку поставщики аппаратных средств назначают адреса и гарантируют, что каждое аппаратное устройство имеет физический адрес, который больше нигде в мире не повторится. Аппаратные устройства различных изготовителей можно соединять в одну физическую сеть, не опасаясь возникновения конфликтов адресов. Статическая адресация является постоянной, поскольку адрес компьютера не меняется после каждой перезагрузки.

Использование **динамической адресации** имеет два преимущества: для изготовителей аппаратных средств отпадает необходимость согласовывать друг с другом назначаемые адреса, а для сетевых администраторов появляется возможность уменьшить размер адресов. Адрес может стать короче, потому что он должен быть уникальным только в одной локальной сети. Динамическая схема адресации позволяет станциям в одной локальной сети выбирать такие же адреса, как и у станций другой локальной сети. Основной недостаток динамической адресации – это отсутствие постоянства и возможность конфликта адресов. При каждой начальной загрузке компьютер получает новый адрес; другие компьютеры должны узнать этот новый адрес, прежде чем они смогут с ним связаться. Более того, если работа сети при загрузке компьютера временно нарушена, то два компьютера могут выбрать один и тот же физический адрес.

Настраиваемые адреса могут служить компромиссом между статической и динамической схемой. Как и статические, настраиваемые адреса являются постоянными: адрес компьютера остается тем же после каждой перезагрузки. Как и динамические, настраиваемые адреса не должны быть слишком большими, поскольку адрес уникален только в конкретной сети. На практике большинство сетевых администраторов предпочитают присваивать последовательные значения настраиваемых адресов. Первый компьютер, установленный в сети, получает адрес 1, второму присваивается адрес 2 и т. д. Одно из преимуществ настраиваемой схемы адресации становится очевидным при замене сетевых интерфейсных аппаратных средств: в отличие от аппаратных средств, в которых используется статически присвоенный адрес, настраиваемый интерфейс может быть заменен без смены физического адреса компьютера.

ШИРОКОВЕЩАТЕЛЬНАЯ РАССЫЛКА

Во многих приложениях, использующих сеть, применяется метод, называемый **широковещательной рассылкой**. Этот термин, который первоначально распространялся на радио- и телевизионные передачи, обозначает такие передачи, которые доступны широкому кругу пользователей. Если приложение выполняет широковещательную рассылку данных, оно предоставляет копию данных в распоряжение всех других компьютеров в сети.

Для обеспечения широковещательной рассылки в локальных сетях используется расширенная схема адресации. Проектировщики сети не только предусматривают присвоение адреса каждому компьютеру, но и назначают специальный зарезервированный адрес, называемый **широковещательным адресом**. Аппаратный интерфейс компьютера настроен на распознавание не только физического адреса станции, но и специального широковещательного адреса. Получив фрейм с одним из этих адресов в поле адреса назначения, интерфейс принимает фрейм и передает его копию операционной системе компьютера.

ГРУППОВАЯ РАССЫЛКА

Групповая рассылка – это ограниченная форма широковещательной рассылки, преимущество которой в том, что в ней для исследования фреймов используются сетевые интерфейсные аппаратные средства. В отличие от широковещательного групповой фрейм не перенаправляется автоматически операционной системе (сетевым протоколам, лежащим выше канального уровня). В интерфейсных аппаратных средствах программируются критерии, в соответствии с которыми они принимают одни групповые фреймы и отбрасывают другие. Решение принимают интерфейсные аппаратные средства и передают выше по стеку только фреймы, соответствующие критериям.

Групповая рассылка предусматривает расширение схемы адресации путем резервирования адресов для групповой рассылки. Во время начальной загрузки компьютера интерфейс программируется на распознавание только адреса компьютера и широковещательного адреса. Если работающее на компьютере приложение должно получать групповые фреймы, оно должно указать сетевому интерфейсу

используемый адрес групповой рассылки. Интерфейс добавляет этот адрес к набору распознаваемых адресов и начинает принимать фреймы, отправленные по этому адресу.

ОПРЕДЕЛЕНИЕ СОДЕРЖИМОГО ФРЕЙМА

Хотя рассматриваемые выше схемы адресации позволяют отправителю указать получателя кадра, используемый при этом адрес не позволяет дать описание содержимого фрейма. Поскольку многие элементы данных имеют одинаковое представление, получатель не может использовать данные кадра, чтобы определить его содержимое. Например, во всех пакетах, которые инкапсулированы во фреймы и содержат сообщения электронной почты, текстовые файлы и Web-страницы, для представления данных используется кодировка ASCII. Для передачи получателю сведений о его содержимом каждый фрейм включает дополнительную информацию, которая указывает тип его информационного наполнения. Для обозначения содержимого фрейма используются следующие методы.

- **Явное обозначение типа фрейма.** Проектировщики сетевого аппаратного обеспечения указывают, как должна быть включена во фрейм информация типа и какие значения должны использоваться для описания фреймов различных типов. Участок фрейма, используемый для описания его содержимого, называют полем типа фрейма, а сам фрейм – автоматически распознаваемым.
- **Неявное обозначение типа фрейма.** При использовании этого метода сетевые аппаратные средства не включают поле типа в каждый фрейм. Фрейм содержит только данные. Поэтому отправитель и получатель должны согласовывать между собой содержимое каждого фрейма или предусматривать использование части данных фрейма в качестве поля типа.

ЗАГОЛОВКИ ФРЕЙМА И ЕГО ФОРМАТ

В локальной сети обычно определен точный формат применяемого в ней фрейма. Хотя отдельные детали могут отличаться, в большинстве сетевых технологий фрейм состоит из двух частей: заголовка фрейма, содержащего адрес источника и адрес назначения, и основной части, называемой телом фрейма, или областью данных,

содержащей передаваемую информацию. Общий формат фрейма показан на рис. 5.1.

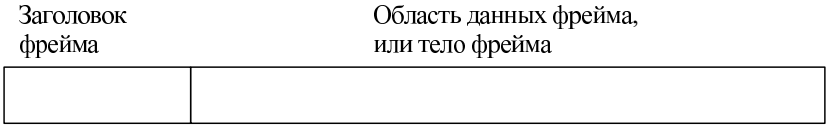


Рис. 5.1. Общий формат фрейма

В большинстве сетевых технологий каждое поле заголовка фрейма имеет постоянный размер и расположение. В результате все фреймы имеют одинаковый размер заголовка. В отличие от этого область данных фрейма не имеет постоянного размера: размер области данных определяется объемом передаваемых данных.

ФОРМАТ ФРЕЙМА ETHERNET

Фрейм Ethernet начинается с заголовка, который включает три поля. 64-битовая начальная серия (преамбула), которая предшествует фрейму, с чередующимися битами 1 и 0, позволяющими аппаратным средствам получателя синхронизировать работу в соответствии с входящим сигналом. Первые два поля заголовка содержат физические адреса. В сети Ethernet используется 48-битовая статическая схема адресации, в которой каждому сетевому интерфейсу (сетевой плате) изготовителем назначается уникальный адрес. Поле, с адресом назначения, содержит физический адрес станции, на которую отправлен данный фрейм. Поле с адресом источника содержит физический адрес станции – отправителя фрейма. Третье поле заголовка включает 16-битовое обозначение **типа фрейма Ethernet или его длину** (рис. 5.2).

Рис. 5.2. Формат фрейма, используемого в сети Ethernet. Число, приведенное в каждом поле, обозначает размер поля в 8-битовых октетах

Фрейм DIX (Digital-Intel-Xerox) сети Ethernet определяет значения, которые могут применяться в полях заголовка, и их смысл. Например, в стандарте указано, что адрес со всеми 48 битами, установленными в значение 1, зарезервирован для широковещательной рассылки, прочие адреса, которые начинаются с бита 1, используются для групповой рассылки, а шестнадцатеричное значение 8137 в поле типа фрейма указывает, что данные во фрейме соответствуют протоколу корпорации Novell, известному под названием IPX. В табл. 5.1 приведены некоторые примеры значений типов Ethernet.

Таблица 5.1

Типы фреймов, используемых в сети Ethernet

Значение	Описание
0000-05DC	Зарезервировано для использования во фреймах LLC/SNAP IEEE
0800	Версия 4 протокола IP Internet
0805	Протокол X.25 CCITT
0900	Сетевой отладчик корпорации Ungermann-Bass
0BAD	Протокол VINES корпорации Banyan Systems
1000-100F	Инкапсуляция концевика пакета Berkeley UNIX
6004	Протокол LAT корпорации Digital Equipment
6559	Протокол Frame Relay
8005	Сетевой зонд корпорации Hewlett Packard
8008	Зарезервировано корпорацией AT&T
8014	Сетевые игры корпорации Silicon Graphics
8035	Протокол Интернет определения доменного имени по сетевому адресу
8038	Оборудование LANBridge корпорации Digital Equipment
805C	Ядро версии V Станфордского университета
809B	Протокол AppleTalk корпорации Apple Computer
80C4-80C5	Зарезервировано корпорацией Banyan Systems
80D5	Протокол SNA корпорации IBM
80FF-8103	Зарезервировано компанией Wellfleet Communications
8137-8138	Протокол IPX корпорации Novell
818D	Зарезервировано корпорацией Motorola
FFFF	Зарезервировано

Как показано в таблице, некоторые типы Ethernet предназначены для использования в системах, разработанных отдельными компаниями, а другие применяются в программном обеспечении, соответствующем международным стандартам, таким как X.25. Наличие стандартизованных обозначений типов гарантирует использование во всех продуктах Ethernet одного и того же значения для конкретного типа фрейма. Поэтому продукты Ethernet одного и того же типа, создаваемые несколькими поставщиками, могут взаимодействовать.

ФРЕЙМЫ, НЕ ОБЕСПЕЧИВАЮЩИЕ АВТОМАТИЧЕСКОГО РАСПОЗНАВАНИЯ ТИПА

Некоторые сетевые технологии не предусматривают в заголовке фрейма поле типа. Это значит, что фреймы не обеспечивают автоматического распознавания типа. Для определения типа данных, содержащихся во фрейме, применяют следующий подход. Перед передачей каких-либо данных отправитель и получатель согласовывают применение первых нескольких октетов поля данных для хранения сведений о типе. Программное обеспечение компьютера-отправителя добавляет эти сведения о типе данных в начало исходящего фрейма. Программное обеспечение в компьютере-получателе извлекает информацию о типе и использует ее при обработке данных.

Для того чтобы во всем программном обеспечении для определения типов применялись одинаковые значения, смысл каждого обозначения типа был установлен организациями по стандартизации. Распределением этих обозначений занималось много организаций, которые не всегда координировали свои действия. Для решения проблемы классификации типов, назначенных разными организациями по стандартизации, институт IEEE разработал стандарт, который предусматривает включение не только поля для обозначения типа, присвоенного организацией, но и поля с указанием самой организации по стандартизации. Эта спецификация, составляющая часть стандарта 802.2 IEEE, известна под названием заголовка управления логическим соединением (LLC – Logical Link Control) стандартного протокола доступа к сети (SNAP – Standard Network Access Protocol). Формат заголовка LLC/SNAP IEEE общепринятый.

На рис. 5.3 приведен пример заголовка LLC/SNAP, который содержит восемь октетов. Первые три октета представляют часть LLC, которая указывает, что далее следует поле типа данных.

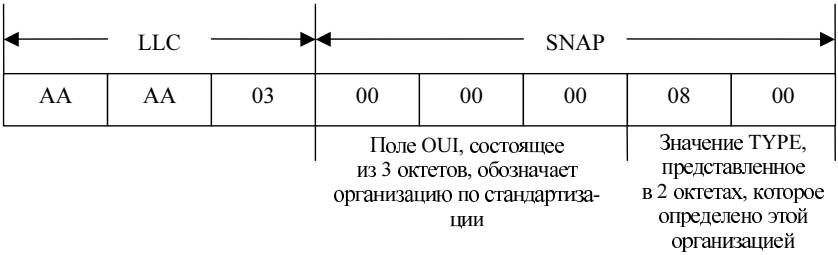


Рис. 5.3. Пример восьмиоктетного заголовка LLC/SNAP IEEE, который используется для указания типа данных. Часть SNAP обозначает организацию и тип, определенный этой организацией

Как показано на рисунке, часть заголовка SNAP разделена на два поля. Первое поле носит название уникального идентификатора организации (OUI – Organizationally Unique Identifier) и используется для обозначения организации по стандартизации. Второе поле содержит значение типа, определенное этой организацией. Например, значение OUI со всеми нулями, показанное на рис. 5.3, принадлежит организации, которая определяет типы Ethernet. Поэтому шестнадцатеричное значение 0800, показанное в этом примере в поле типа, интерпретируется в соответствии со стандартом, в котором определены типы Ethernet. Как и типы, закодированные в заголовке фрейма, поле типа LLC/SNAP дает возможность всем компьютерам совместно использовать сеть для широковещательной рассылки фреймов. При поступлении фрейма на компьютер проверяется информация LLC/SNAP в начале области данных фрейма. Если получатель не распознает OUI или не имеет программного обеспечения для обработки данных полученного типа, фрейм отбрасывается. Поэтому широковещательный фрейм, несущий данные определенного типа, будет проигнорирован всеми компьютерами сети, кроме тех, которые распознают этот тип данных.

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

Кадры представлены в виде файлов (Y:/redmy/DNL/stud_ether/ver2) двоичного формата (отсутствуют преамбула и контрольная сумма, для исходящего кадра длина может быть меньше минимальной). В программе должна быть предусмотрена возможность выбора файла.

Минимальная информация, которую должна выдавать программа включает в себя:

1) количество фреймов в файле;
2) тип каждого фрейма;
3) IP- адреса (основную информацию заголовка IP-пакета);
4) MAC- адреса (основную информацию заголовка кадра).

ВАРИАНТЫ ЗАДАНИЙ

1. Программа должна выполнить анализ файла с именем ethernetnn, где nn – номер вашей бригады.
2. Программа должна выполнить полный анализ фрейма с номером nn, где nn – номер вашей бригады.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Как осуществляется передача фреймов по сети?
2. Какие способы адресации вы знаете? В чем они заключаются? Приведите достоинства и недостатки каждого из способов.
3. Что называется широковещательной рассылкой? В каких случаях она используется? В чем ее достоинства и недостатки?
4. Что называется групповой рассылкой? В каких случаях она используется? Чем она отличается от широковещательной рассылки?
5. Какие методы используются для обозначения содержимого фрейма?
6. Приведите общий формат фрейма, передаваемого по сети.
7. Приведите форматы фреймов Ethernet.
8. Укажите несколько типов фреймов, используемых в сети Ethernet.
9. Как определяют тип данных, содержащихся во фрейме, не обеспечивающем автоматическое распознавание типа?

Лабораторная работа № 6
ДИАГНОСТИКА IP ПРОТОКОЛА

Цель работы. Выполнить анализ основных утилит мониторинга сети (ping, traceroute и др.). Спроектировать и реализовать программу, выполняющую основные функции утилиты ping/traceroute.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ И МЕТОДИЧЕСКИЕ УКАЗАНИЯ

При работе в Интернете время от времени возникают ситуации, когда нужно определить, работоспособен ли тот или иной канал или узел, а в случае работы с динамическими протоколами маршрутизации выяснить, по какому из каналов вы в данный момент работаете. Используется эта процедура и для оценки вероятности потери пакетов в заданных сегментах сети или каналах. Для решения этих задач удобна программа **Ping** (программа эхо-тестирования). Вызывая эту программу, пользователь должен задать параметр с указанием имени или числового адреса удаленного компьютера в сети. Программа **ping** посылает сообщение на указанный компьютер, а затем в течение небольшого промежутка времени ожидает ответа. При поступлении ответа программа **ping** сообщает пользователю, что компьютер является действующим; иначе сообщает, что компьютер не отвечает. Некоторые версии программы **ping** дают пользователю возможность указывать размер отправляемого пакета, вычисляют время кругового обращения пакета (т. е. время с момента отправки сообщения до момента получения ответа).

Программа **ping** широко применяется на практике в качестве диагностического инструментального средства, несмотря на то что на первый взгляд кажется, что программа слишком проста, чтобы быть

полезной. При обнаружении неисправности в сети с помощью программы **ping** можно определить, какие части сети работают правильно и где возникла неисправность. Полученные результаты позволяют быстро найти место отказа.

Ping – хорошее средство проверки правильности конфигурации сети, поскольку в выполнении этой команды участвуют система маршрутизации, схемы разрешения адресов и сетевые шлюзы. Если данная команда не работает, можете быть совершенно уверены, что более сложные средства тем более не функционируют. Несмотря на свою простоту, **ping** – одна из главных рабочих лошадок, используемых при отладке сетей.

Программа **ping** служит для принудительного вызова ответа конкретной машины. Для этого используется дейтаграмма ECHO_ REQUEST протокола ICMP. Это протокол низкого уровня, который не требует наличия серверных процессов на зондируемой машине; это хороший способ убедиться в том, что питание машины включено и IP находится в поднятом состоянии. Успешный результат использования утилиты **ping** вовсе не означает, что выполняются какие-то сервисные программы высокого уровня.

Протокол ICMP изначально рассматривался как способ передачи отправителю сообщения о невозможности доставки IP-дейтаграммы. Позднее были разработаны перспективные способы использования созданной системы управления сообщениями. В частности, на ее основе были разработаны инструментальные средства, предназначенные для сбора информации об объединенной сети посредством отправки таких дейтаграмм, которые позволяют выявлять ошибки.

ОСНОВЫ ПРОТОКОЛА ICMP

В сети могут возникать такие нарушения в работе, информация о которых должна быть доставлена всем участникам обмена данными. Набор протоколов TCP/IP включает протокол, используемый протоколом IP для формирования сообщений об ошибках при возникновении нарушений: ICMP (Internet Control Message Protocol). Протоколы IP и ICMP зависят друг от друга. Протокол IP использует ICMP для формирования сообщений об ошибках, а протокол ICMP использует IP для доставки сообщений (рис. 6.1).

Рис. 6.1. Формат ICMP-сообщения

Поле «необязательные данные» имеет переменную длину и содержит данные, которые необходимо вернуть отправителю. Поля «идентификатор» и «номер» используются отправителем для проверки соответствия между запросом и ответом. В табл. 6.1 приведены сообщения протокола ICMP.

Таблица 6.1

Сообщения протокола ICMP

Тип	Значение
0	Ответ эхоповтора
1	Не присвоен
2	Не присвоен
3	Получатель недоступен
4	Подавление источника
5	Перенаправление
6	Альтернативный адрес хоста
7	Не присвоен
8	Запрос эхоповтора
9	Объявление маршрутизатора
10	Выбор маршрутизатора
11	Истечение тайм-аута
12	Ошибка в параметре

Окончание табл. 6.1

Тип	Значение
13	Запрос отметки времени
14	Ответ на запрос отметки времени
15	Информационный запрос
16	Информационный ответ
17	Запрос маски адреса
18	Ответ на запрос маски адреса
19	Зарезервирован (для применения в средствах защиты)
20–29	Зарезервирован (для испытаний на устойчивость)
30	Сообщение утилиты traceroute
31	Ошибка преобразования дейтаграммы
32	Перенаправление мобильного хоста
33	Сообщение Where-Are-You IPv6
34	Сообщение I-Am-Here IPv6
35	Запрос регистрации мобильного хоста
36	Ответ на запрос регистрации мобильного хоста
37–255	Зарезервировано

Ниже приведены примеры сообщений протокола ICMP об ошибках.

- *Получатель недоступен.* Обнаружив, что дейтаграмма не может быть доставлена в место назначения, маршрутизатор отправляет хосту, на котором она была создана, сообщение о том, что получатель недоступен. В сообщении указано, является ли недоступным конкретный хост назначения или недоступна сеть, к которой подключен хост назначения.
- *Запрос/ответ эхоповтора.* Запрос эхоповтора может быть передан программному обеспечению протокола ICMP любого компьютера. В ответ на запрос эхоповтора программное обеспечение ICMP должно послать ответ эхоповтора ICMP. Ответ содержит те же данные, что и запрос.

Таким образом, в программе ***ping*** используются запросы и ответы эхоповтора ICMP.

После вызова на выполнение программа **ping** отправляет по указанному адресу назначения дейтаграмму IP, которая содержит запрос

эхоповтора ICMP. Отправив запрос, программа в течение короткого времени ожидает ответа. Если ответ не поступает, программа **ping** повторяет запрос. Если ответ не поступает после нескольких попыток передачи (или приходит сообщение ICMP о том, что место назначения недостижимо), программа **ping** выдает сообщение, что удаленный компьютер недоступен. На запрос эхоповтора отвечает программное обеспечение ICMP удаленного компьютера в соответствии с протоколом, при получении запроса эхоповтора программное обеспечение ICMP должно отправить ответ. Взаимодействие утилит **Ping** и **Tracert** (Traceroune) со стеком протоколов TCP/IP показано на рис. 6.2.

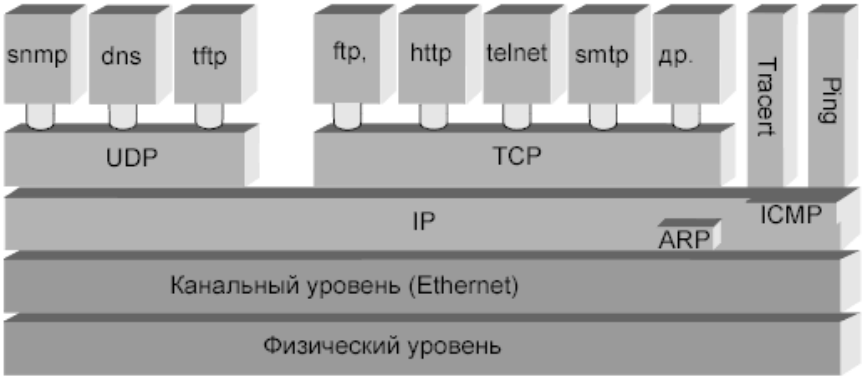


Рис. 6.2. Основные протоколы стека TCP/IP

РЕАЛИЗАЦИЯ

Для практической реализации, как всегда, можно выбрать несколько подходов. Первый из них – использование низкоуровневых функций API (встроенных в библиотеку ICMP.DLL). Второй – использование высокоуровневых компонентов (к примеру, Indy IdICMPClient).

И у первого, и у второго способа есть свои позитивные и негативные моменты. Так, при выборе функций API откомпилированный код будет иметь гораздо меньшие размеры, чем при высокоуровневых компонентах, да и производительность его будет выше (например, при одновременном пинге одной подсети с использованием потоков).

С другой стороны, компоненты можно использовать, имея только отдаленное представление о работе с протоколом ICMP, а также об

Windows API. Но в то же время компоненты порождают неоправданно большой исполняемый код, да и производительность в этом случае ниже.

РЕАЛИЗАЦИЯ С ИСПОЛЬЗОВАНИЕМ WINDOWS API

При использовании Windows API для написания функции пинга используется библиотека ICMP (icmp.dll), которая предоставляет интерфейс для работы с одноименным протоколом. В этой библиотеке реализованы следующие функции:

```
• function IcmpCreateFile() : THandle; stdcall;
external 'ICMP.DLL' name 'IcmpCreateFile';
        //создает соединение
• function IcmpCloseHandle(IcmpHandle : THandle) :
BOOL; stdcall; external 'ICMP.DLL' name 'IcmpClose-
Handle';   //закрывает соединение
• function IcmpSendEcho(
IcmpHandle : THandle; // handle, возвращенный Icmp-
CreateFile()
DestAddress : u_long; // адрес получателя (в сете-
вом порядке)
RequestData : PVOID; // указатель на посылаемые
данные
RequestSize : Word; // размер посылаемых данных
RequestOptns : PIPINFO; // указатель на посылаемую
структуру
// ip_option_information (может быть nil)
ReplyBuffer : PVOID; // указатель на буфер, содер-
жащий ответы.
ReplySize : DWORD; // размер буфера ответов
Timeout : DWORD // время ожидания ответа в миллисе-
кундах
) : DWORD; stdcall; external 'ICMP.DLL' name
'IcmpSendEcho';
// посылает ICMP эхо-запрос по заданному IP адресу
и помещает все
// ответы, полученные за время заданного таймаута,
в буфер ответов.
```

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

В лабораторной работе требуется реализовать приложение, которое будет выполнять основные функции одной из утилит мониторинга сети (например, ping).

ВАРИАНТЫ ЗАДАНИЙ

1. Реализовать функцию «Запрос эхоповтора». В поле данных необходимо поместить текст по указанию преподавателя.
2. Реализовать функцию «Запрос отметки времени».
3. Реализовать функцию «Запрос маски адреса».

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Как определить доступность рабочей станции в сети Интернет?
2. Как определить количество маршрутизаторов на пути от вашего компьютера до требуемого вам Web-сайта?
3. Назовите основные типы ICMP-сообщений.
4. Назначение поля «Код» в ICMP-сообщении.
5. Основные возможности стандартной утилиты ping.
6. Модель реализации утилиты traceroute.
7. Структура ICMP-пакета.
8. Схема инкапсуляции ICMP-пакета в Ethernet-кадр.
9. Размер поля данных ICMP-пакета.

Лабораторная работа № 7

АНАЛИЗ ОСНОВНЫХ СТАНДАРТОВ БЕСПРОВОДНОЙ СВЯЗИ

Цель работы. Выполнить анализ основных принципов работы беспроводных сетей (стандартов IEEE 802.11(a,b,g,n), IEEE 802.16(d,e,m), IEEE 802.15). Приобрести практические навыки работы в беспроводной сети ФПМИ.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ И МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Здесь кратко рассматриваются основные стандарты беспроводной технологии – IEEE 802.11, IEEE 802.16, IEEE 802.15 и их расширения (например, IEEE 802.11a, IEEE 802.11b, IEEE 802.11g, IEEE 802.11n), работающие на частоте 2,4 и 5,0 ГГц. Стандарты IEEE 802.11a, IEEE 802.11b, IEEE 802.11g, IEEE 802.11n на сегодняшний день являются наиболее популярным стандартами Radio Ethernet.

1. СРАВНЕНИЕ ТЕХНОЛОГИЙ БЕСПРОВОДНОЙ СВЯЗИ

Сравнение основных стандартов беспроводной связи показано в табл. 7.1.

Сопоставление WiMAX и Wi-Fi далеко не редкость – термины созвучны, названия стандартов, на которых основаны эти технологии, похожи (стандарты разработаны IEEE, оба начинаются с «802.»), а также обе технологии основаны на беспроводных соединениях и используются для подключения к Интернету (каналу обмена данными). Но несмотря на это *эти технологии направлены на решение различных задач*.

1.1. WiMAX

WiMAX – это система дальнего действия, покрывающая километры пространства, которая обычно использует лицензированные спектры частот (хотя возможно и использование нелицензированных частот) для предоставления провайдером соединения с Интернетом типа точка-точка конечному пользователю. Разные стандарты семейства 802.16 обеспечивают разные виды доступа, от мобильного (похож на передачу данных с мобильных телефонов) до фиксированного (альтернатива проводному доступу, при котором беспроводное оборудование пользователя привязано к местоположению).

Таблица 7.1

Сравнительная таблица стандартов беспроводной связи

Технология	Стандарт	Использование	Пропускная способность	Радиус действия	Частоты
Wi-Fi	802.11a	WLAN	до 54 Мбит/с	До 100 м	5,0 ГГц
Wi-Fi	802.11b	WLAN	до 11 Мбит/с	До 100 м	2,4 ГГц
Wi-Fi	802.11g	WLAN	до 54 Мбит/с	До 100 м	2,4 ГГц
Wi-Fi	802.11n	WLAN	до 300 Мбит/с (в перспективе до 450, а затем до 600 Мбит/с)	До 100 м	2,4…2,5 или 5,0 ГГц
WiMAX	802.16d	WMAN	до 75 Мбит/с	6…10 км	1,5…11 ГГц
WiMAX	802.16e	Mobile WMAN	до 40 Мбит/с	1…5 км	2,3…13,6 ГГц
WiMAX	802.16m	WMAN, Mobile WMAN	до 1 Гбит/с (WMAN), до 100 Мбит/с (Mobile WMAN)	н/д (стандарт в разработке)	н/д (стандарт в разработке)
Bluetooth v. 1.1	802.15.1	WPAN	до 0,7 Мбит/с	До 10 м	2,4 ГГц
Bluetooth v. 2.0	802.15.3	WPAN	до 3 Мбит/с	До 100 м	2,4 ГГц
Bluetooth v. 3.0	802.11	WPAN	от 3 Мбит/с до 24 Мбит/с	До 100 м	2,4 ГГц
UWB	802.15.3a	WPAN	110…480 Мбит/с	До 10 м	3,1…10,6 ГГц

Окончание табл. 7.1

Технология	Стандарт	Использование	Пропускная способность	Радиус действия	Частоты
ZigBee	802.15.4	WPAN	от 20 до 250 Кбит/с	1…100 м	2,4 ГГц (16 каналов), 915 МГц (10 каналов), 868 МГц (один канал)
Инфракрасный порт	IrDa	WPAN	до 16 Мбит/с	от 5 до 50 сантиметров, односторонняя связь – до 10 метров	

1.2. Wi-Fi

Wi-Fi – это система более короткого действия, обычно покрывающая десятки метров, которая использует нелицензированные диапазоны частот для обеспечения доступа к сети. Обычно Wi-Fi применяется пользователями для доступа к их собственной локальной сети, которая может быть и не подключена к Интернету. Если WiMAX можно сравнить с мобильной связью, то Wi-Fi скорее похож на стационарный беспроводной телефон.

WiMAX и Wi-Fi имеют совершенно разный механизм Quality of Service (QoS). WiMAX использует механизм, основанный на установлении соединения между базовой станцией и устройством пользователя. Каждое соединение основано на специальном алгоритме планирования, который обеспечивает параметр QoS для каждого соединения. Wi-Fi, в свою очередь, использует механизм QoS, подобный тому, что используется в Ethernet, при котором пакеты получают различный приоритет. Такой подход не гарантирует одинакового QoS для каждого соединения.

На рис. 7.1 показаны характеристики по дальности и скорости для основных технологий беспроводного доступа.

Существуют различные подходы к классификации беспроводных технологий.

- По дальности действия:
  - беспроводные персональные сети (WPAN – Wireless Personal Area Networks). Примеры технологий – Bluetooth;

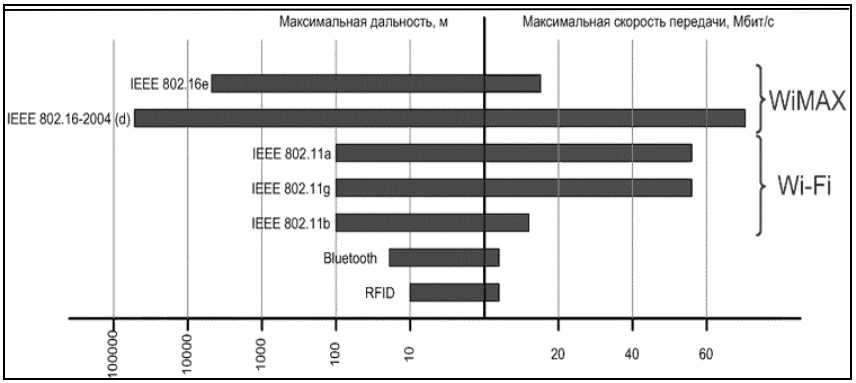


Рис. 7.1. Сравнение основных технологий беспроводного доступа по дальности и скорости

- беспроводные локальные сети (WLAN – Wireless Local Area Networks). Примеры технологий – Wi-Fi;
- беспроводные сети масштаба города (WMAN – Wireless Metropolitan Area Networks). Примеры технологий – WiMAX;
- беспроводные глобальные сети (WWAN – Wireless Wide Area Network). Примеры технологий – CSD, GPRS, EDGE, EV-DO, HSPA;

Классификация беспроводных технологий по дальности действия приведена на рис. 7.2.

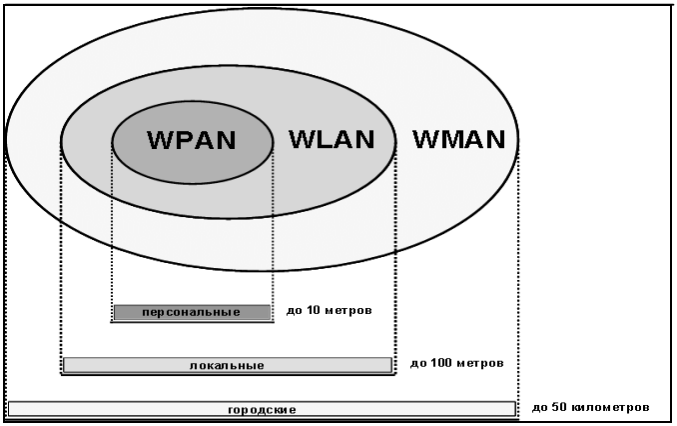


Рис. 7.2. Классификация беспроводных технологий по дальности действия

По скорости и мобильности – на рис. 7.3.

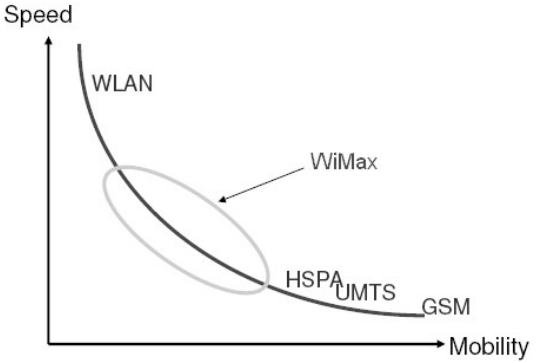


Рис. 7.3. Мобильность и скорость WiMAX по сравнению с другими беспроводными технологиями:

HSPA – технология беспроводной широкополосной радиосвязи; UMTS – технология сотовой связи, разработанная Европейским Институтом Стандартов Телекоммуникаций (ETSI) для внедрения 3G в Европе (подчеркивается преемственность в разработках с сетями стандарта GSM)

1.3. Wi-Fi И ТЕЛЕФОНЫ СОТОВОЙ СВЯЗИ

Некоторые считают, что Wi-Fi и подобные ему технологии со временем могут заменить сотовые сети, такие как GSM. Препятствиями для такого развития событий в ближайшем будущем служат отсутствие глобального роуминга, ограниченность частотного диапазона и сильно ограниченный радиус действия Wi-Fi. Более правильным выглядит сравнение сотовых сетей с другими стандартами беспроводных сетей, таких как UMTS, CDMA или WiMAX.

Тем не менее Wi-Fi пригоден для использования VoIP в корпоративных сетях или в среде SOHO. Первые образцы оборудования появились уже в начале 2000-х, однако на рынок они вышли только в 2005 году. Тогда же такие компании, как Zyxel, UT Starcomm, Samsung, Hitachi и многие другие, представили на рынок VoIP Wi-Fi-телефоны по «разумным» ценам. В 2005 году ADSL ISP провайдеры начали предоставлять услуги VoIP своим клиентам. Когда звонки с помощью VoIP стали очень дешевыми, а зачастую вообще бесплатными, провайдеры, способные предоставлять услуги VoIP, получили воз-

можность открыть новый рынок – услуг VoIP. Телефоны GSM с интегрированной поддержкой возможностей Wi-Fi и VoIP начали выводиться на рынок, и потенциально они могут заменить проводные телефоны. В настоящий момент непосредственное сравнение Wi-Fi и сотовых сетей нецелесообразно. Телефоны, использующие только Wi-Fi, имеют очень ограниченный радиус действия, поэтому развертывание таких сетей обходится очень дорого. Тем не менее оно может быть наилучшим решением для локального использования, например, в корпоративных сетях. Стоит заметить, что при наличии в данном конкретном месте покрытия как GSM, так и Wi-Fi экономически намного выгоднее использовать Wi-Fi, разговаривая посредством сервисов Интернет-телефонии. Например, клиент Skype давно существует в версиях как для смартфонов, так и для КПК.

2. РАЗВИТИЕ ТЕХНОЛОГИИ БЕСПРОВОДНЫХ СЕТЕЙ

2.1. СТАНДАРТ IEEE 802.11

Комитет по стандартам IEEE 802 сформировал рабочую группу для беспроводных локальных сетей 802.11 в 1990 году. Эта группа занялась разработкой всеобщего стандарта для радиооборудования и сетей, работающих на частоте 2,4 ГГц, со скоростями доступа 1 и 2 Mbps (Megabits per second – мегабит в секунду). Работы по созданию стандарта были завершены через 7 лет, и в июне 1997 года была ратифицирована первая спецификация 802.11. Стандарт IEEE 802.11 стал первым стандартом для продуктов WLAN (Wireless LAN – беспроводная локальная сеть) от независимой международной организации, разрабатывающей большинство стандартов для проводных сетей.

Как и все стандарты IEEE 802, 802.11 работает на нижних двух уровнях модели ISO/OSI, физическом уровне и канальном уровне (рис. 7.4). Любое сетевое приложение, сетевая операционная система или протокол (например, TCP) будут так же хорошо работать в сети 802.11, как и в сети Ethernet.

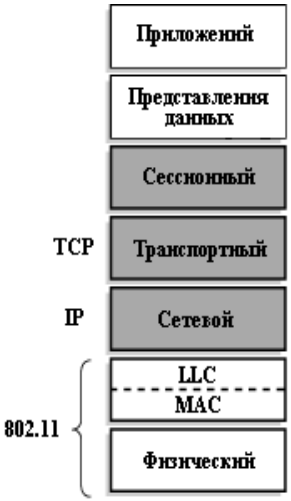


Рис. 7.4. Уровни модели ISO/OSI и их соответствие стандарту 802.11

Основная архитектура, особенности и службы следующих расширений стандартов определяются в первоначальном стандарте 802.11. Спецификации этих расширений затрагивают только **физический уровень**, добавляя лишь более высокие скорости доступа.

2.2. ФИЗИЧЕСКИЙ УРОВЕНЬ 802.11

На физическом уровне определены два широкополосных радиочастотных метода передачи и один в инфракрасном диапазоне. Радиочастотные методы работают в диапазоне 2,4 ГГц и обычно используют полосу 83 МГц от 2,400 до 2,483 ГГц. Технологии *широкополосного сигнала*, используемые в радиочастотных методах, увеличивают надежность, пропускную способность, позволяют многим не связанным друг с другом устройствам разделять одну полосу частот с минимальными помехами друг для друга.

Стандарт 802.11 использует метод прямой последовательности (Direct Sequence Spread Spectrum – **DSSS**) и метод частотных скачков (Frequency Hopping Spread Spectrum – **FHSS**). Эти методы кардинально отличаются и несовместимы друг с другом.

Для модуляции сигнала FHSS использует технологию Frequency Shift Keying (FSK). При работе на скорости 1 Мбит/с используется FSK модуляция по Гауссу второго уровня, а при работе на скорости 2 Мбит/с – четвертого уровня.

Метод DSSS использует технологию модуляции Phase Shift Keying (PSK). При этом на скорости 1 Мбит/с будет дифференциальная двоичная PSK, а на скорости 2 Мбит/с – дифференциальная квадратичная PSK модуляция.

Заголовки физического уровня всегда передаются на скорости 1 Мбит/с, в то время как данные могут передаваться со скоростями 1 и 2 Мбит/с.

2.2.1. Передача с расширением спектра методом частотных скачков (FHSS)

При использовании метода частотных скачков полоса 2,4 ГГц разделяется на 79 каналов по 1 МГц. Отправитель и получатель согласовывают схему переключения каналов (на выбор имеется 22 такие схемы), и данные посылаются последовательно по различным каналам с использованием этой схемы. Каждая передача данных в сети 802.11 происходит по разным схемам переключения, а сами схемы разработа-

ны таким образом, чтобы минимизировать вероятность того, что два отправителя будут использовать один и тот же канал одновременно.

Метод FHSS позволяет использовать очень простую схему приемопередатчика, однако ограничен максимальной скоростью 2 Мбит/с. Такое ограничение вызвано тем, что под один канал выделяется ровно 1 МГц, и это вынуждает FHSS-системы использовать весь диапазон 2,4 ГГц. Поэтому необходимо частое переключение каналов, что, в свою очередь, приводит к увеличению накладных расходов.

2.2.2. Передача с расширением спектра методом прямой последовательности (DSSS)

Метод DSSS делит диапазон 2,4 ГГц на 14 частично перекрывающихся каналов. Для того чтобы несколько каналов могли использоваться одновременно в одном и том же месте, необходимо, чтобы они отстояли друг от друга на 25 МГц (не перекрывались) для исключения взаимных помех. Таким образом, в одном месте может одновременно использоваться максимум 3 канала. Данные пересылаются с использованием одного из этих каналов без переключения на другие каналы. Чтобы компенсировать посторонние шумы, используется 11-битовая последовательность Баркера, когда каждый бит данных пользователя преобразуется в 11 бит передаваемых данных. Такая высокая избыточность для каждого бита позволяет существенно повысить надежность передачи, при этом значительно снизив мощность передаваемого сигнала. Даже если часть сигнала будет утеряна, он в большинстве случаев все равно будет восстановлен. Тем самым минимизируется число повторных передач данных.

2.3. КАНАЛЬНЫЙ УРОВЕНЬ 802.11

Канальный уровень IEEE 802.11 состоит из двух подуровней: управления логической связью (Logical Link Control, LLC) и управления доступом к носителю (Media Access Control, MAC). Стандарт 802.11 использует тот же LLC и 48-битовую адресацию, как и другие сети 802, что позволяет легко объединять беспроводные и проводные сети, однако MAC-уровень имеет кардинальные отличия.

MAC-уровень 802.11 очень похож на реализованный в 802.3, где он поддерживает множество пользователей на общем носителе (разделяемая среда передачи), когда пользователь проверяет носитель перед доступом к нему. Для Ethernet сетей 802.3 используется протокол

Carrier Sence Multiple Access with Collision Detection – CSMA/CD (множественный доступ с контролем несущей и определением коллизий). CSMA/CD определяет, как станции Ethernet получают доступ к проводной линии и как они обнаруживают и обрабатывают коллизии, возникающие в том случае, если несколько устройств пытаются одновременно установить связь по сети. Чтобы обнаружить коллизию, станция должна обладать способностью и *принимать, и передавать одновременно,* т. е. применять дуплексные приемопередатчики. Стандарт 802.11 предусматривает использование полудуплексных приемопередатчиков, поэтому в беспроводных сетях 802.11 станция не может обнаружить коллизию во время передачи.

Чтобы учесть это отличие, 802.11 предусматривается модифицированный протокол, известный как Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA), или Distributed Coordination Function (DCF). CSMA/CA (множественный доступ с контролем несущей и предотвращением коллизий) пытается избежать коллизий путем явного подтверждения фрейма (ACK), что означает, что принимающая станция посылает ACK-пакет для подтверждения того, что пакет получен неповрежденным.

CSMA/CA работает следующим образом. Станция, желающая передавать, тестирует канал, и если не обнаружено активности, ожидает в течение некоторого случайного промежутка времени, а затем передает, если среда передачи данных все еще свободна. Если пакет приходит целым, принимающая станция посылает пакет ACK, прием которого отправителем завершает процесс передачи. Если передающая станция не получила пакет ACK в силу того, что не был получен пакет данных, или пришел поврежденный ACK, делается предположение, что произошла коллизия, и пакет данных передается снова через случайный промежуток времени.

Для определения того, свободен ли канал, используется алгоритм оценки чистоты канала (Channel Clearance Algorithm, CCA). Его суть заключается в измерении энергии сигнала на антенне и определения мощности принятого сигнала (RSSI). Если мощность принятого сигнала ниже заданного порога, то канал объявляется свободным и MAC-уровень получает статус CTS. Если мощность выше порогового значения, передача данных задерживается в соответствии с правилами протокола. Стандарт предоставляет еще одну возможность определения незанятости канала, которая может использоваться либо отдельно, либо вместе с измерением RSSI-метода проверки несущей. Этот метод

выборочный, так как с его помощью определяется тип несущей, что и по спецификации 802.11. Наилучший метод для использования зависит от того, каков уровень помех в рабочей области.

Таким образом, CSMA/CA предоставляет способ разделения доступа по радиоканалу. Механизм явного подтверждения эффективно решает проблемы помех. Однако он добавляет некоторые дополнительные накладные расходы, которых нет в 802.3, поэтому сети 802.11 будут всегда работать медленнее, чем эквивалентные им Ethernet локальные сети.

Другая специфичная проблема MAC-уровня – это проблема «скрытой точки», когда обе станции могут «слышать» точку доступа, но не могут «слышать» друг друга в силу большого расстояния или преград (рис. 7.5). Для решения этой проблемы в 802.11 на MAC-уровне добавлен необязательный протокол Request to Send/Clear to Send (RTS/CTS). Когда используется этот протокол, посылающая станция передает RTS и ждет ответа точки доступа с CTS. Поскольку все станции в сети могут «слышать» точку доступа, сигнал CTS заставляет их отложить свои передачи, что позволяет передающей станции передать данные и получить ACK-пакет без возможности коллизий. Так как RTS/CTS добавляет дополнительные накладные расходы на сеть, временно резервируя носитель, он обычно используется только для пакетов очень большого объема, для которых повторная передача была бы слишком дорогостоящей.

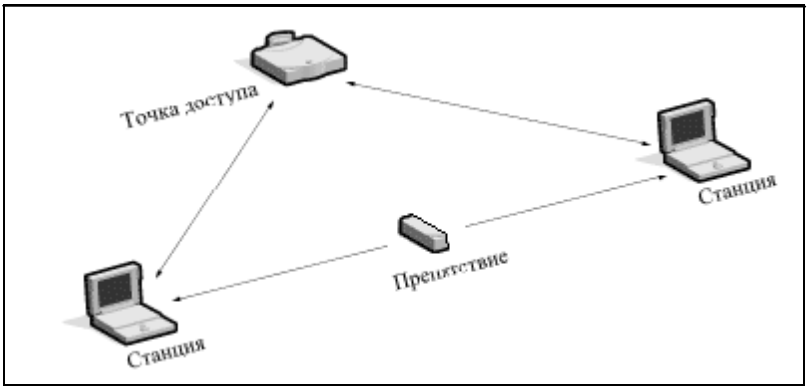


Рис. 7.5. Иллюстрация проблемы «скрытой точки»

Наконец, MAC-уровень 802.11 предоставляет возможность расчета CRC и фрагментации фреймов. Каждый фрейм имеет свою контрольную сумму CRC, которая рассчитывается и прикрепляется к фрейму. Здесь наблюдается отличие от сетей Ethernet, в которых обработкой ошибок занимаются протоколы более высокого уровня (например, TCP). Фрагментация фреймов позволяет разбивать большие фреймы на более маленькие при передаче по радиоканалу, что полезно в очень «заселенных» средах или в тех случаях, когда существуют значительные помехи, так как у меньших фреймов меньше вероятности быть поврежденными. Этот метод в большинстве случаев уменьшает необходимость повторной передачи и, таким образом, увеличивает производительность всей беспроводной сети. MAC-уровень ответствен за сборку полученных фрагментов, делая этот процесс «прозрачным» для протоколов более высокого уровня.

2.3.1. Подключение к сети

MAC-уровень 802.11 несет ответственность за то, каким образом клиент подключается к точке доступа. Когда клиент 802.11 попадает в зону действия одной или нескольких точек доступа, он на основе мощности сигнала и наблюдаемого значения количества ошибок выбирает одну из них и подключается к ней. Как только клиент получает подтверждение того, что он принят точкой доступа, он настраивается на радиоканал, в котором она работает. Время от времени он проверяет все каналы 802.11, чтобы посмотреть, не предоставляет ли другая точка доступа службы более высокого качества. Если такая точка доступа находится, то станция подключается к ней, перенастраиваясь на ее частоту (рис. 7.6).

Переподключение обычно происходит в том случае, если станция была физически перемещена вдаль от точки доступа, что вызвало ослабление сигнала. В других случаях повторное подключение происходит из-за изменения радиочастотных характеристик здания или просто из-за большого сетевого трафика через первоначальную точку доступа. В последнем случае эта функция протокола известна как «балансировка нагрузки», так как ее главное назначение – наиболее эффективно распределить общую нагрузку на беспроводную сеть по всей доступной инфраструктуре сети.

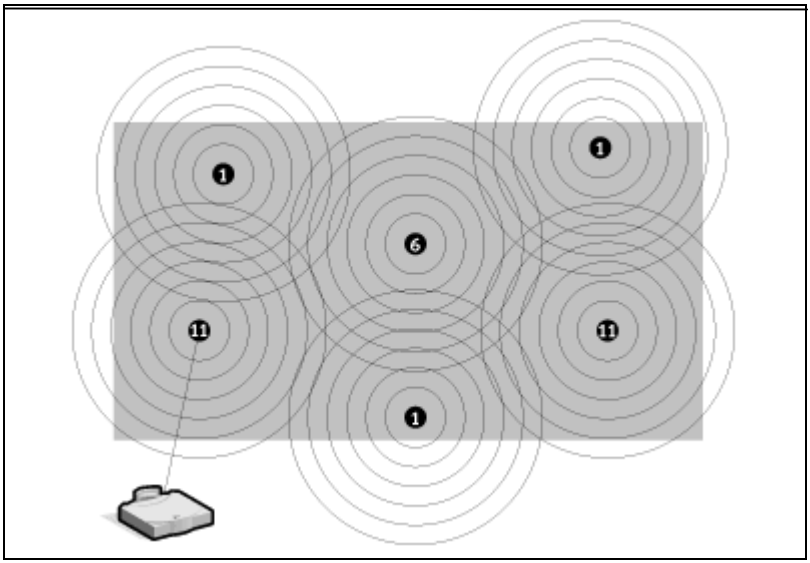


Рис. 7.6. Подключение к сети и иллюстрация правильного назначения каналов для точек доступа

Процесс динамического подключения и переподключения позволяет сетевым администраторам устанавливать беспроводные сети с очень широким покрытием, создавая частично перекрывающиеся «соты». Идеальным вариантом будет такой, при котором соседние перекрывающиеся точки доступа станут использовать разные DSSS каналы, чтобы не создавать помех в работе друг другу (рис. 7.6).

2.3.2. Поддержка потоковых данных

Потоковые данные, такие как видео- или голос, поддерживаются в спецификации 802.11 на MAC-уровне посредством Point Coordination Function (PCF). В противоположность Distributed Coordination Function (DCF), где управление распределено между всеми станциями, в режиме PCF только точка доступа управляет доступом к каналу. В том случае, если установлен BSS (подсистема базовых станций) с включенной PCF, время равномерно распределяется промежутками для работы в режиме PCF и в режиме CSMA/CA. Во время периодов, когда система находится в режиме PCF, точка доступа опрашивает все станции для

получения данных. На каждую станцию выделяется фиксированный промежуток времени, по истечении которого производится опрос следующей станции. Ни одна из станций не может передавать в это время, за исключением той, которая опрашивается. Так как PCF дает возможность каждой станции передавать в определенное время, то гарантируется максимальная латентность. Недостаток такой схемы заключается в том, что точка доступа должна производить опрос всех станций, а это становится чрезвычайно неэффективным в больших сетях.

2.3.3. Режимы работы 802.11

Стандарт IEEE 802.11 определяет два типа оборудования – клиент (Supplicant), который укомплектован беспроводным сетевым интерфейсом, и точку доступа (Access Point, AP), которая выполняет роль моста между беспроводной и проводной сетью. Точка доступа обычно содержит в себе приемопередатчик, интерфейс проводной сети (802.3), а также программное обеспечение, занимающееся обработкой данных.

Стандарт IEEE 802.11 определяет два режима работы сети: режим «Ad-hoc» и режим инфраструктуры (infrastructure mode – клиент-сервер). В режиме клиент/сервер (рис. 7.7) беспроводная сеть состоит как минимум из одной точки доступа, подключенной к проводной сети, и некоторого набора беспроводных оконечных станций. Такая конфигурация носит название базового набора служб (Basic Service Set, BSS).

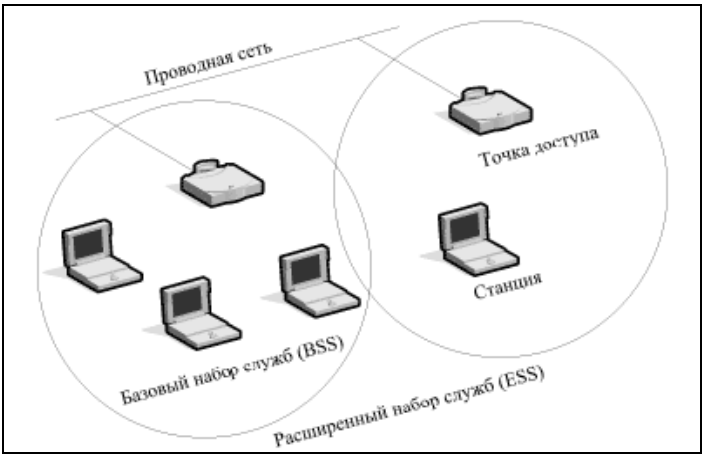


Рис. 7.7. Архитектура сети «клиент/сервер»

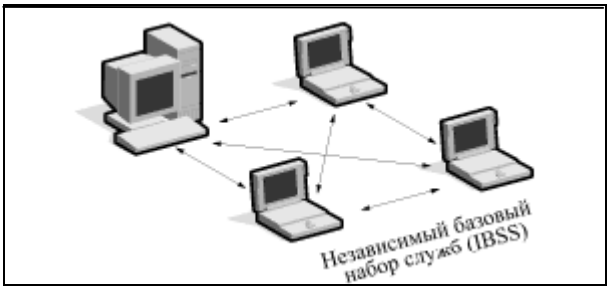


Рис. 7.8. Архитектура сети «Ad-hoc»

Два или более BSS, образующих единую подсеть, формируют расширенный набор служб (Extended Service Set, ESS). Так как большинству беспроводных станций требуется получать доступ к файловым серверам, принтерам, Интернету, доступным в проводной локальной сети, они будут работать в режиме клиент/сервер.

Режим «Ad-hoc» (также называемый точка-точка, или независимый базовый набор служб, IBSS) – это простая сеть, в которой связь между многочисленными станциями устанавливается напрямую, без использования специальной точки доступа (рис. 7.8). Такой режим полезен в том случае, если инфраструктура беспроводной сети не сформирована (например, отель, выставочный зал, аэропорт) либо по каким-то причинам не может быть сформирована.

3. РАСШИРЕНИЕ СТАНДАРТА 802.11

Заложенная первоначально скорость передачи данных в беспроводной сети (1 и 2 Мбит/с) не удовлетворяла потребностям пользователей. Для того чтобы сделать технологию Wireless LAN популярной, дешевой, а главное, удовлетворяющей современным жестким требованиям бизнес-приложений, разработчики были вынуждены создать новые расширения стандарта (IEEE 802.11a/b/g/n). Основные изменения, внесенные в IEEE 802.11, – это поддержка новых более высоких скоростей передачи данных. Для достижения этих скоростей был выбран метод DSSS, так как метод частотных скачков в силу ограничений FCC не может поддерживать более высокие скорости. Из этого следует, что системы 802.11b/g будут совместимы с DSSS-системами 802.11, но не будут работать с системами FHSS 802.11.

3.1. IEEE 802.11A

Стандарт сетей Wi-Fi использует частотный диапазон 5 ГГц. Несмотря на то что эта версия применяется не так часто из-за стандартизации IEEE 802.11b и внедрения 802.11g, она также претерпела изменения в плане частоты и модуляции (OFDM). OFDM – *это механизм мультиплексирования* (*уплотнения*) *посредством ортогональных поднесущих, который позволяет передавать данные параллельно на множественных подчастотах.* Это позволяет повысить устойчивость к помехам и, поскольку отправляется более одного потока данных, реализуется высокая пропускная способность. IEEE 802.11a может развивать скорость вплоть до 54 Мбит/с в идеальных условиях. В менее идеальных условиях устройства могут вести связь со скоростью 48, 36, 24, 18, 12 и 6 Мбит/с. Стандарт IEEE 802.11a несовместим с 802.11b 802.11g.

3.2. IEEE 802.11B/G

В сентябре 1999 года IEEE ратифицировал расширение предыдущего стандарта. Названное IEEE 802.11b (также известное как 802.11 High rate), оно определяет стандарт для продуктов беспроводных сетей, которые работают на скорости 11 Мбит/с (подобно Ethernet), что позволяет успешно применять эти устройства в крупных организациях. Совместимость продуктов различных производителей гарантируется независимой организацией, которая называется Wireless Ethernet Compatibility Alliance (WECA). Эта организация была создана лидерами индустрии беспроводной связи в 1999 году. В настоящее время членами WECA являются более 80 компаний, в том числе такие известные производители, как Cisco, Lucent, IBM, Intel, Apple, Dell, Fujitsu, Siemens, Sony, AMD и пр.

Основные изменения, внесенные в IEEE 802.11, – это поддержка двух новых скоростей передачи данных – 5,5 и 11 Мбит/с. Для достижения этих скоростей был выбран метод DSSS, так как метод частотных скачков в силу ограничений FCC не может поддерживать более высокие скорости.

Для поддержки очень зашумленных сред, а также работы на больших расстояниях сети 802.11b/g используют динамический сдвиг скорости, который позволяет автоматически изменять скорость передачи данных в зависимости от свойств радиоканала. Например, пользователь может подключиться с максимальной скоростью 11 Мбит/с, но в том случае, если повысится уровень помех или пользователь удалится

на большое расстояние, мобильное устройство начнет передавать на меньшей скорости – 5,5, 2 или 1 Мбит/с. В том случае, если возможна устойчивая работа на более высокой скорости, мобильное устройство автоматически начнет передавать с более высокой скоростью. Сдвиг скорости – механизм физического уровня, является прозрачным для вышестоящих уровней и пользователя.

3.3. IEEE 802.11N

Версия стандарта 802.11 для сетей Wi-Fi повышает скорость передачи данных практически вчетверо по сравнению с устройствами стандартов 802.11g (максимальная скорость которых равна 54 Мбит/с), при условии использования в режиме 802.11n с другими устройствами 802.11n. Теоретически 802.11n способен обеспечить скорость передачи данных до 600 Мбит/с, применяя передачу данных сразу по четырем антеннам. По одной антенне – до 150 Мбит/с. Устройства 802.11n работают в диапазонах 2,4…2,5 или 5,0 ГГц. Кроме того, устройства 802.11n могут работать в трех режимах:

- наследуемом (Legacy), в котором обеспечивается поддержка устройств 802.11b/g и 802.11a;
- смешанном (Mixed), в котором поддерживаются устройства 802.11b/g, 802.11a и 802.11n;
- «чистом» режиме – 802.11n (именно в этом режиме и можно воспользоваться преимуществами повышенной скорости и увеличенной дальностью передачи данных, обеспечиваемыми стандартом 802.11n).

Черновую версию стандарта 802.11n (DRAFT 2.0) поддерживают многие современные сетевые устройства. Итоговая версия стандарта (DRAFT 11.0), которая была принята 11 сентября 2009 года, обеспечивает скорость до 300 Мбит/с, многоканальный вход/выход, известный как **MIMO**, и большее покрытие. На 2011 год имелось небольшое количество устройств, соответствующих финальному стандарту. Например, у компании D-LINK основная продукция проходила стандартизацию в 2008 году. Существуют компании, занимающиеся перестандартизацией основной продукции.

3.3.1. ОСОБЕННОСТИ СТАНДАРТА

Реальная скорость передачи данных

Реальная скорость передачи данных всегда меньше канальной скорости. Для Wi-Fi реальная скорость передачи данных обычно отличается более чем в 2 раза в меньшую сторону.

Кроме того, существует еще несколько факторов, ограничивающих реальную пропускную способность:

- канал всегда делится между клиентами;
- передавая служебный трафик, точка доступа всегда подстраивается под клиента, работающего на минимальной скорости;
- наличие помех (работающие рядом точки доступа, микроволновые печи, «радионяни», bluetooth-устройства, радиотелефоны).

Стоит отметить, что при работе в стандарте 802.11b или при обеспечении совместимого с ним режима существует всего три непересекающихся канала, т. е. которые не мешают друг другу (обычно это 1-й, 6-й и 11-й). Следовательно, если у соседа за стеной работает точка доступа на 1-м канале, а у вас дома на 3-м, то эти точки доступа будут мешать друг другу, тем самым уменьшая скорость передачи данных.

Два частотных диапазона

Устройства стандарта 802.11n могут работать в одном из двух диапазонов – 2,4 или 5 ГГц. Это намного повышает гибкость их применения, позволяя отстраиваться от источников радиочастотных помех. При выборе подходящей системы ИТ-специалистам следует иметь в виду, что практически все клиенты 802.11n на основе **CardBus** и **ExpressCard** пока рассчитаны только на диапазон 2,4 ГГц, но несколько встраиваемых адаптеров и плат типоразмера **mini-PCI** способны поддерживать оба.

Каналы шириной 40 MHz

Спецификация 802.11n предусматривает использование как стандартных каналов шириной 20 МГц, так и широкополосных – на 40 МГц, в результате чего повышается пропускная способность. При этом в диапазоне 2,4 ГГц умещается всего два непересекающихся канала удвоенной ширины.

MIMO

Ключевой компонент стандарта 802.11n под названием *MIMO* (Multiple Input, Multiple Output – много входов, много выходов) предусматривает применение пространственного мультиплексирования для одновременной передачи нескольких информационных потоков по одному каналу, а также многолучевое отражение, которое обеспечивает доставку каждого бита информации соответствующему получателю с небольшой вероятностью влияния помех и потерь данных. Именно возможность одновременной передачи и приема данных определяет высокую пропускную способность устройств 802.11n.

На начало 2013 года большинство предлагаемых производителями точек доступа поддерживает MIMO 2×2 или 1×1, т. е. SISO (однопотоковая передача). Встроенные в мобильные устройства Wi-Fi-адаптеры обычно поддерживают режим SISO.

3.3.2. АНТЕННЫ

Чаще всего стандартными считаются антенные конфигурации цепи для передачи и приема информации 3×3 или 2×3, однако со временем устройства стандарта 802.11n станут поддерживать и другие варианты. В простых недорогих моделях будет реализована схема из одной передающей и двух принимающих цепей (по статистике абоненты потребляют гораздо больше данных, чем передают), тогда как пользователи, которым нужна очень большая скорость передачи данных, смогут приобрести старшие модели с конфигурацией антенн 4×4.

4. ПОСТРОЕНИЕ WLAN

Задача создания корпоративной беспроводной сети не так проста, и прежде чем перейти к ее решению, необходимо разобраться в некоторых особенностях и инструментах ***беспроводного строительства***.

Что нужно учитывать, при построении WLAN? Наиболее распространенными стандартами беспроводных сетей сегодня являются IEEE 802.11b и 802.11g. Оборудование таких сетей согласно IEEE работает в диапазоне 2400…2483,5 МГц и способно передавать данные с максимальной скоростью 11 и 54 Мбит/с соответственно. Практически все ноутбуки на платформе Centrino поддерживают указанные стандарты.

Распределение волн в рассматриваемом диапазоне имеет ряд оригинальных качеств. Несмотря на функциональное сходство беспроводного и проводного оборудования, разница в их установке, монтаже и настройке немалая. Причина – в свойствах физических сред, используемых для передачи информации. В случае с беспроводным оборудованием нужно учитывать законы распространения радиоволн. Радиоэфир более чувствителен к различного рода помехам. Поэтому наличие перегородок, стен и железобетонных перекрытий может сказаться на скорости передачи данных. Условия приема и передачи радиосигнала ухудшают не только физические препятствия, помехи создают и различные радиоизлучающие приборы. При разрешении таких проблем одним только правилом прямолинейного распространения радиоволн (с эмпирическим определением коэффициента наносимых преградами

помех) не обойтись, ведь выявить тип преграды – тоже задача не из легких.

Проблема качества сигнала не решится простым увеличением мощности точек доступа. Дело в том, что такой подход не гарантирует повышения качества связи, а скорее, наоборот, ведет к его ухудшению, так как создает массу помех в том диапазоне частот, который используют другие точки доступа. Напомним, что точки доступа 802.11 предоставляют пользователям среду, в которой в определенный момент времени лишь одна из них может вести передачу данных. Как следствие, масштабирование таких сетей ограничено.

Стандартно точки доступа комплектуются всенаправленными антеннами, часто приходится выбирать между качественным сигналом и уровнем доступности сети за пределами офиса (т. е. безопасностью): ведь доступ к среде, по которой передаются данные, открыт. Именно поэтому вопросы безопасности сети, построенной на основе нескольких точек доступа, весьма актуальны.

4.1. АРХИТЕКТУРЫ

Перейдем к выбору архитектуры создаваемой сети. Распределенная или централизованная? Каждая из них имеет ряд особенностей, достоинств и недостатков. Так, например, для построения сети на основе распределенной архитектуры (distributed access point architecture) достаточно установить точки доступа. Это, несомненно, ее преимущество. Дело в том, что стандарт 802.11 изначально объединяет в одном устройстве функциональность сетевого контроллера и радиотрансиверов, поэтому развернуть сеть можно посредством установки точек доступа в свободный порт коммутатора и беспроводных адаптеров в клиентские ПК. В большинстве случаев нет даже необходимости конфигурировать точки доступа или клиентские компьютеры, поэтому беспроводной сегмент становится естественной частью всей сети.

Существенный недостаток такой сети – отсутствие единого управляющего элемента. Поэтому применение способа без настройки построения зачастую сильно ограничено.

Проблема распределенного построения сети решается использованием беспроводных коммутаторов, однако их применение уже символизирует организацию беспроводной сети на основе централизованной архитектуры. Основное отличие проводных коммутаторов от беспроводных в том, что последние не предоставляют пользователю выде-

ленную полосу пропускания. (Для этого пришлось бы предоставить отдельный беспроводной канал каждому пользователю сети, в таком случае беспроводные сети лишаются главного достоинства.) Имеются и общие черты. Так, в сети, где устанавливается беспроводной коммутатор, функции шифрования и аутентификации от точек доступа переходят к коммутатору и администрируются централизованно. В итоге задача точки доступа ограничивается транзитом данных к пользователю и от него.

4.2. ВОПРОСЫ БЕЗОПАСНОСТИ

Основным фактором, сдерживающим широкое распространение беспроводных сетей в корпоративной среде, служит устоявшееся мнение о недостаточном уровне безопасности таких решений. Впрочем, заключение небезосновательное: базовые средства защиты 802.11 взламывались неоднократно. Сегодня беспроводную сеть считают защищенной, если в ней функционируют три основные составляющие системы безопасности: *аутентификация пользователя, конфиденциальность и целостность передачи данных.*

В настоящее время сообщество разработчиков и производителей беспроводного оборудования близко к повсеместному признанию модели, базирующейся на технологии Wi-Fi Protected Access (WPA). Такая технология поддерживает базовые средства аутентификации протоколов 802.1X, конфиденциальность передачи данных посредством шифрования трафика с помощью TKIP и целостность информации путем сверки контрольной суммы MIC (Message Integrity Check). Отметим, что протокол TKIP (Temporal Key Integrity Protocol) широкого распространения не получил: несмотря на увеличение общего уровня безопасности, он существенно сужает пропускную способность беспроводного канала.

При использовании распределенной архитектуры составляющие WPA-технологии обеспечиваются точками доступа. Иначе обстоит дело в случае централизованной архитектуры. Некоторые производители возлагают задачу защиты сети на точки доступа, другие – на беспроводные коммутаторы, третьи распределяют между этими двумя устройствами. Например, на коммутаторы возлагается аутентификация пользователей, а шифрованием и целостностью данных занимаются точки доступа. Однако правильнее, когда аутентификация выполняется на границе сети: если злоумышленник пытается получить доступ к сети, лучше перехватить его как можно раньше.

Рассмотрим, как проходит атака DoS (Denial of Service). В распределенной архитектуре DoS-атака осуществляется только на одну точку доступа; так как последняя блокирует весь неавторизованный трафик, другие компоненты сети будут изолированы от атаки. В сети с централизованной архитектурой, где функцию аутентификации выполняет коммутатор, точка доступа не может заблокировать неавторизованный трафик, поэтому он направляется к беспроводному коммутатору, который не пропускает его в корпоративную сеть. Однако этот трафик будет передаваться через все устройства, находящиеся на пути от атакованной точки доступа к коммутатору.

Так как при распределенной архитектуре все функции безопасности сосредоточиваются в точке доступа, то наиболее частый аргумент против такой архитектуры – это ее физическая уязвимость. Иными словами, к точке доступа очень просто несанкционированно проникнуть или вообще похитить ее, а это грозит серьезными проблемами: именно в ней содержатся ключи шифрования и другие установки по обеспечению безопасности. Злоумышленник, похитивший точку доступа, может затем извлечь из нее весьма важную информацию, включая MAC-адрес других сетевых устройств. Более того, установив украденную точку доступа в своей сети (достаточно близко от атакуемой), он может перехватить полноправного клиента и раскрыть регистрационную информацию. Впрочем, проблему можно предотвратить: помещение, где налаживается беспроводная сеть, должно иметь хорошую охранную систему, а точки доступа следует располагать на максимально возможной высоте (производственные корпуса). В иных случаях для достижения достаточного уровня безопасности сети лучше остановиться на централизованной архитектуре.

Не последнее место среди оборудования, повышающего уровень безопасности беспроводной сети, занимают антенны. Если учесть, что беспроводные клиентские адаптеры поставляются со стандартными антеннами, поддерживающими относительно небольшую дальность передачи данных, важность использования направленных антенн для физического ограничения зоны доступа к сети становится весьма актуальной.

Решение проблемы безопасности во многом должно продвинуть распространение протокола передачи данных, заложенного в спецификацию 802.11i. Последняя предполагает внесение изменений не только в протокол, но и в стандартную аппаратуру приемопередающих устройств. Предлагаемый к использованию протокол аутентификации

Extensible Authentication Protocol (EAP) весьма эффективен, так как помимо стойких алгоритмов шифрования предлагает еще и применение 128-битовых ключей. Кроме того, процедура аутентификации предполагает участие в ней трех сторон: вызывающей (клиента), вызываемой (точки доступа) и сервера аутентификации, что существенно повышает безопасность соединения.

4.2.1. Развертывание системы сетевой безопасности WPA

Через «дыры» в системе защиты беспроводных ЛВС (БЛВС), основанной на протоколе WEP (Wired Equivalent Privacy) комитета 802.11, может, образно выражаясь, проехать грузовик. К счастью, появились новые технологии обеспечения информационной безопасности БЛВС, которые описаны в стандартах 802.11i института IEEE и WPA (Wi-Fi Protected Access) организации Wi-Fi Alliance и призваны «залатать» эти «дыры».

Технология WPA пришла на замену технологии WEP. Плюсами WPA являются усиленная безопасность данных и ужесточенный контроль доступа к беспроводным сетям. Немаловажной характеристикой является совместимость между множеством беспроводных устройств как на аппаратном уровне, так и на программном. Сегодня WPA и WPA2 разрабатываются и продвигаются организацией Wi-Fi Alliance.

Структуру защитной технологии WPA можно наглядно выразить с помощью следующей формулы (представленной в виде операторов языка программирования):

$$WPA = \{802.1X + EAP + TKIP + MIC + (RADIUS^{*}y)\},$$

где $y = 0$, *If PSK* (*используется упрощенный режим*); *ELSE* $y = 1$.

Из этой формулы видно, что WPA, по сути, – это сумма нескольких технологий. В стандарте WPA используется расширяемый протокол аутентификации (EAP) как основа для механизма аутентификации пользователей. Непременным условием аутентификации считается предъявление пользователем свидетельства (иначе называют мандатом), подтверждающего его право на доступ в сеть. Для этого права пользователь проходит проверку по специальной базе зарегистрированных пользователей. Без аутентификации работа в сети для пользователя будет запрещена. База зарегистрированных пользователей и система проверки в больших сетях, как правило, расположены на специальном сервере (чаще всего RADIUS).

Следует отметить, что WPA имеет упрощенный режим. Он получил название Pre-Shared Key (WPA-PSK). При применении режима PSK необходимо ввести один пароль для каждого отдельного узла беспроводной сети. Если пароли совпадают с записями в базе, пользователь получит разрешение на доступ в сеть.

В стандарте WPA предусмотрено использование защитных протоколов 802.1X, EAP, TKIP и RADIUS. Механизм аутентификации пользователей основан на протоколах 802.1X и EAP (Extensible Authentication Protocol). EAP позволяет сетевому администратору задействовать множество алгоритмов аутентификации пользователей посредством сервера RADIUS.

Подобно WPA, 802.11i имеет режим pre-shared key (режим PSK, известен также как персональный режим), предназначенный для применения дома и в небольших офисных сетях, которые не могут себе позволить использования аутентификационного сервера 802.1X. Каждый пользователь для получения доступа в сеть должен ввести пароль. Пароль обычно запоминается в машине пользователя, так что его достаточно ввести лишь один раз. Пароли должны иметь не менее 8 символов, однако рекомендуется 20 символов. Стандарт IEEE 802.11i допускает более строгие PSK, содержащие до 63 шестнадцатеричных символов.

Стандарт IEEE 802.11i, известен под названием WPA2 (Wi-Fi Protected Access), служит дополнением к стандарту 802.11 и специфицирует механизмы обеспечения безопасности для беспроводных сетей. Проект стандарта был принят 24 июня 2004 года и заменяет предыдущую спецификацию WEP (Wired Equivalent Privacy), которая имела определенные слабости. Алгоритм защищенного доступа Wi-Fi (WPA) был разработан альянсом Wi-Fi в качестве промежуточного решения проблемы безопасности. WPA использует подмножество регламентаций 802.11i. Альянс Wi-Fi одобрил регламентации WPA2 как полностью совместимые с 802.11i. Стандарт 802.11i использует блочный шифр AES(Advanced Encryption Standard); WEP и WPA используют поточный шифр RC4.

Архитектура 802.11i содержит следующие компоненты: 802.1X для аутентификации, RSN (Robust Security Network) для отслеживания ассоциаций и CCMP (Counter Mode with Cipher Block Chaining Message Authentication Protocol), базирующийся на алгоритме шифрования AES. Другим важным элементом процесса аутентификации является четырехходовой диалог.

Единственное отличие WPA-PSK состоит в том, что аутентификация производится с использованием пароля, а не по сертификату пользователя.

Если в вашей организации имеется множество устаревших клиентских станций и точек доступа стандарта 802.11b, то заменить сразу все эти устройства на новые практически нереально. В этом случае лучше ориентироваться на стандарт WPA, который представляет собой подмножество спецификаций стандарта 802.11i. Эти два стандарта совместимы друг с другом, поэтому использование поддерживающих WPA-продуктов можно считать начальным этапом перехода к системе защиты на базе стандарта 802.11i.

4.3. ВЫВОДЫ

Беспроводные локальные сети имеют как преимущества, так и недостатки в сравнении с проводными локальными сетями. Если необходимо быстро и/или ненадолго развернуть локальную сеть, то предпочтение отдается беспроводной технологии. Примером этому могут служить различного рода конференции, семинары и т. п. Удобно также использовать беспроводные технологии на предприятиях или в аэропортах, где необходимо обслуживать клиентов, которые все время перемещаются. К недостаткам беспроводных сетей можно отнести низкую пропускную способность и относительную небезопасность. Сети Radio Ethernet всегда работают медленнее, чем эквивалентные Ethernet сети (из-за механизма явного подтверждения CSMA/CA).

Стандарт IEEE 802.11i (WPA2) специфицирует механизмы обеспечения безопасности для беспроводных сетей. Согласно этому стандарту беспроводную сеть считают защищенной, если в ней функционируют три основные составляющие системы безопасности: *аутентификация пользователя, конфиденциальность и целостность передачи данных.*

Любая услуга должна быть безопасной. В настоящей момент единственным действующим сертификатом по безопасности Wi-Fi является WPA2.

5. АППАРАТНОЕ ОБЕСПЕЧЕНИЕ БЕСПРОВОДНОГО ДОСТУПА В ЛОКАЛЬНОЙ СЕТИ ФПМИ

Локальная сеть ФПМИ обеспечена необходимым оборудованием для использования услуг беспроводного доступа. На рис. 7.9 показано устройство DAP-1360, на базе которого реализован беспроводный

доступ в локальной сети факультета ФПМИ. В сети факультета ФПМИ используется несколько таких устройств (пока четыре).

Рис. 7.9. Устройство DAP-1360

Устройство DAP-1360 представляет собой беспроводную точку доступа с поддержкой режима маршрутизатора. Используя DAP-1360, можно быстро организовать беспроводную сеть и разрешить сотрудникам и студентам ФПМИ подключаться к ней практически в любой точке (в зоне действия беспроводной сети). Точка доступа может выполнять функции базовой станции для подключения к беспроводной сети устройств, работающих по стандартам 802.11b, 802.11g и 802.11n (со скоростью до 300 Мбит/с).

В устройстве реализовано множество функций для беспроводного интерфейса. Устройство поддерживает несколько стандартов безопасности (WEP, WPA/WPA2), фильтрацию подключаемых устройств по MAC-адресу, несколько режимов работы (**точка доступа, маршрутизатор, клиент**), а также позволяет использовать технологии WPS, WDS и WMM.

WPS (Wi-Fi Protected Setup) – это технология, предназначенная для быстрой и безопасной установки беспроводной сети. В отличие от выполнения традиционных настроек WPS автоматически обозначает имя сети и задает шифрование для защиты от несанкционированного доступа в сеть. При этом не требуется специальных знаний и нет необходимости настраивать оборудование, вручную задавая все параметры.

WDS (Wireless Distribution System) – технология, позволяющая расширить зону покрытия беспроводной сети посредством объединения нескольких WiFi-точек доступа в единую сеть без необходимости наличия проводного соединения между ними (что является обязательным при традиционной схеме построения сети). Отличительная черта технологии по сравнению с другими решениями – сохранение MAC-адресов клиентов сети.

WMM (Wireless MultiMedia). В отличие от 802.1p, где определено восемь классов трафика, в WMM их всего четыре: Voice, Video, Background и Best Effort. Голосовой трафик (AC_VO) имеет наивысший приоритет.

В режиме маршрутизатора устройство DAP-1360 оснащено встроенным межсетевым экраном. Расширенные функции безопасности позволяют минимизировать последствия действий хакеров и предотвращают вторжение в вашу сеть и доступ к нежелательным сайтам для пользователей вашей локальной сети. Для управления и настройки DAP-1360 используется простой и удобный встроенный web-интерфейс.

В локальной сети ФПМИ аутентификация клиентов (Supplicant) осуществляется по учетным записям Active Directory (AD) с использованием сервера (Authentication server) RADIUS. Схема аутентификации показана на рис. 7.10.

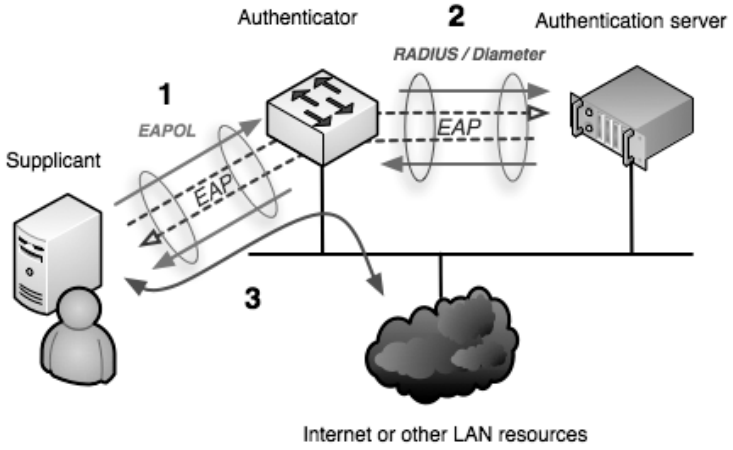


Рис. 7.10. Схема аутентификации для беспроводной сети ФПМИ

Следует отметить, что в беспроводной сети ФПМИ реализована распределенная архитектура, что предполагает возможность обеспечивать безопасность средствами точки доступа (база MAC-адресов точки доступа, WEP-шифрование). В случае добавления клиентов нет необходимости перестраивать архитектуру сети.

Таким образом, сеть ФПМИ обеспечена необходимыми условиями для использования услуг беспроводного доступа.

Настройки клиента и сервера для различных ОС

Следует отметить, что настройки клиентского адаптера в разных операционных системах (Windows, UNIX) в общем случае происходят по-разному, но они одинаково прозрачны для пользователя, т. е. настройка происходит автоматически. Применяется множество способов настройки точки доступа из различных операционных систем: Web-интерфейс, telnet и др. Настройка точки доступа в сети ФПМИ осуществлялась из операционной системы Windows через интуитивно понятный Web-интерфейс.

ЗАДАНИЕ К ЛАБОРАТОРНОЙ РАБОТЕ

1. Изучить основы построения беспроводных сетей.
2. Провести сравнение безопасности беспроводной и проводной локальных сетей.
3. Определить необходимые условия для использования услуг беспроводного доступа (оборудование на клиенте и сервере).
4. Научиться создавать беспроводное соединение в режиме Ad-hoc на базе доступного оборудования стандарта 802.11.
5. Научиться настраивать персональный компьютер для управления точкой доступа.
6. Выполнить анализ инфраструктурного режима сетей Wi-Fi.
7. Ответить на контрольные вопросы.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назовите и дайте краткую характеристику основным технологиям беспроводного доступа.
2. Приведите классификацию технологий беспроводного доступа по дальности действия.

3. Каким образом гарантируется совместимость беспроводных продуктов различных производителей?

4. Назовите все уровни модели ISO/OSI.

5. На каких уровнях модели ISO/OSI работает 802.11?

6. Что означает «соединение в режиме Ad-hoc»?

7. Какие технологии широкополосного доступа использует стандарт 802.11?

8. Какая модуляция используется в стандарте 802.11?

9. Какие режимы работы сети определяет стандарт 802.11?

10. На какой радиочастотный диапазон ориентирован стандарт 802.11b?

11. Какая скорость передачи будет предельной для стандарта 802.11a?

12. Перечислите способы реализации канала беспроводной сети 802.11.

13. Какой метод доступа к среде передачи использует MAC-уровень 802.11?

14. Дайте характеристику проблеме «скрытой точки».

15. Что позволяет сетевым администраторам устанавливать беспроводные сети с очень широким покрытием?

16. Назовите основные особенности стандарта 802.11n.

17. Что необходимо учитывать при построении WLAN?

18. Основные составляющие системы безопасности беспроводных сетей и их краткая характеристика.

19. Дайте характеристику режимам WPA.

20. Формула защитной технологии WPA.

21. Какие защитные протоколы используются в стандарте 802.11i?

22. Перечислите основные методы обеспечения информационной безопасности беспроводных сетей.

23. От чего зависит скорость соединения ТД и беспроводного клиента?

24. Какой протокол обеспечивает более высокий уровень информационной безопасности WEP или WPA ?

25. Назовите основное отличие протокола WPA2 от протокола WPA.

26. Какое максимальное количество каналов сети 802.11b может одновременно использоваться в одной точке пространства?

27. К чему необходимо стремиться при проектировании размещения точек доступа?

28. С помощью чего осуществляется управление точками доступа?

29. Какой сети должен принадлежать адрес вашего компьютера, чтобы иметь возможность управлять точкой доступа?

30. Для чего нужен роуминг?

31. Для чего служит точка доступа, работающая в инфраструктурном режиме?

32. Какие режимы работы поддерживает DAP-1360?

33. Смогут ли другие беспроводные адаптеры подключаться к точке, работающей в Wireless Client режиме?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. *Камер Д.* Компьютерные сети и Internet. Разработка приложений для Internet: пер. с англ. – М.: Издательский дом «Вильямс», 2002. – 640 с.

2. *Куроуз Дж., Росс К.* Компьютерные сети. Многоуровневая архитектура Интернета. – 2-е изд. – СПб.: Питер, 2004. – 765 с.

3. Основы WEB-технологий/ П.Б. Храмцов и др. – М.: ИНТУИТ.РУ «Интернет-университет информационных технологий», 2003. – 512 с.

4. *Семенов Ю.А.* (ГНЦ ИТЭФ), book.itep.ru Телекоммуникационные технологии. http://www.citforum.ru

5. *Долозов Н.Л.* Сетевые информационные технологии. – Ч. 1: учеб. пособие / Н.Л. Долозов. – Новосибирск: Изд-во НГТУ, 2009. – 100 с.

6. *Беспроводные* технологии. http://citforum.ru/nets/wireless.

Приложение А

ПРИМЕР API-ИНТЕРФЕЙСА ПРИКЛАДНОГО ПРОГРАММИРОВАНИЯ ДЛЯ ОБМЕНА ДАННЫМИ ПО СЕТИ.

А1. ПРИМЕР КОДА ЭХО-СЕРВЕРА

server.cpp

```
#pragma comment (lib,«Ws2_32.lib»)
#include <winsock2.h>
#include <stdio.h>
#include <iostream>
#include <sstream>
#include <string>
using namespace std;
int main(void)
{
 WORD sockVer;
 WSADATA wsaData;
 int retVal;
 sockVer = MAKEWORD(2,2);
 WSAStartup(sockVer, &wsaData);
 //Создаем сокет
 SOCKET   servSock   =   socket(PF_INET,   SOCK_STREAM,
IPPROTO_TCP);

 if(servSock == INVALID_SOCKET)
 {
 printf(«Unable to create socket\n»);
 WSACleanup();
            system(«pause»);
 return SOCKET_ERROR;
 }
```

```
      SOCKADDR_IN sin;
 sin.sin_family = PF_INET;
 sin.sin_port = htons(3114);
 sin.sin_addr.s_addr = INADDR_ANY;

 retVal     =     bind(servSock,     (LPSOCKADDR)&sin,
sizeof(sin));
 if(retVal == SOCKET_ERROR)
 {
 printf(«Unable to bind\n»);
 WSACleanup();
           system(«pause»);
 return SOCKET_ERROR;
 }
     printf(«Server   started   at   %s,   port   %d\n»,
inet_ntoa(sin.sin_addr), htons(sin.sin_port));
     while(true)
     {
           //Пытаемся начать слушать сокет
           retVal = listen(servSock, 10);
           if(retVal == SOCKET_ERROR)
           {
                 printf(«Unable to listen\n»);
                 WSACleanup();
                  system(«pause»);
                 return SOCKET_ERROR;
           }
           //Ждем клиента
           SOCKET clientSock;
           SOCKADDR_IN from;
           int fromlen=sizeof(from);
           clientSock   =   accept(servSock,   (struct
sockaddr*)&from, &fromlen);
           if(clientSock == INVALID_SOCKET)
            {
```

```
                printf(«Unable to accept\n»);
                WSACleanup();
                system(«pause»);
                return SOCKET_ERROR;
            }
printf(«New connection accepted from  %s, port  %d\n»,
inet_ntoa(from.sin_addr), htons(from.sin_port)) ;
           char szReq[256];
           retVal = recv(clientSock, szReq, 256, 0);
      //Пытаемся получить данные от клиента
           if(retVal == SOCKET_ERROR)
            {
                printf(«Unable to recv\n»);
                system(«pause»);
                return SOCKET_ERROR;
            }
           printf(«Data received\n»);
           string s = (const char*) szReq;
           if(s[0]=='s')// Команда на выключение серве-
ра
            {
                char *szResp = «Server shutdown»;
                retVal  =  send(clientSock,  szResp,
256, 0);

                closesocket(clientSock);
                break;
            }
           else
            {
               istringstream ss(s);
               int a, b;
               ss >> a;
               ss >> b;
               char szResp[256];
```

```
                sprintf(szResp," %d», a+b);

                //Пытаемся отосладь данные клиенту
                printf(«Sending     response     from
server\n»);
                retVal = send(clientSock, szResp, 256,
0);

                if(retVal == SOCKET_ERROR)
                {
                      printf(«Unable to send\n»);
                      system(«pause»);
                      return SOCKET_ERROR;
                }
            closesocket(clientSock);
                printf(«Connection closed\n»);
       }
      }
      //Закрываем сокет
 closesocket(servSock);
 WSACleanup();
 return 0;
}
```

А2. ПРИМЕР КОДА КЛИЕНТА СЛУЖБЫ ЭХОПОВТОРА

client.cpp

```
#pragma comment (lib,«Ws2_32.lib»)
#include <stdio.h>
#include <winsock2.h>
#include <string>
#include <iostream>
using namespace std;
int main()
{
 WORD ver = MAKEWORD(2,2);
 WSADATA wsaData;
 int retVal=0;
 WSAStartup(ver,(LPWSADATA)&wsaData);
 LPHOSTENT hostEnt;
 hostEnt = gethostbyname(«localhost»);
 if(!hostEnt)
 {
 printf(«Unable to collect gethostbyname\n»);
 WSACleanup();
            system(«pause»);
 return 1;
 }
 //Создаем сокет
 SOCKET   clientSock   =   socket(PF_INET,   SOCK_STREAM,
IPPROTO_TCP);
 if(clientSock == SOCKET_ERROR)
 {
 printf(«Unable to create socket\n»);
 WSACleanup();
            system(«pause»);
 return 1;
 }
 string ip;
 cout << «ip>";
 cin >> ip;
 cin.ignore();
```

```
 SOCKADDR_IN serverInfo;
 serverInfo.sin_family = PF_INET;
serverInfo.sin_addr.S_un.S_addr                          =
inet_addr(ip.c_str());
 serverInfo.sin_port = htons(3114);
 //Пытаемся присоединится к серверу по ip и port
 retVal=connect(clientSock,(LPSOCKADDR)&serverInfo,
sizeof(serverInfo));
 if(retVal==SOCKET_ERROR)
 {
 printf(«Unable to connect\n»);
 WSACleanup();
           system(«pause»);
 return 1;
 }
 printf(«Connection made sucessfully\n»);
      printf(«Enter  a  and  b  or  'stop'  to  shutdown
server\n»);
      char pBuf[256];
      gets(pBuf);
 printf(«Sending request from client\n»);
 //Отсылаем данные на сервер
 retVal = send(clientSock, pBuf, strlen(pBuf), 0);
 if(retVal == SOCKET_ERROR)
 {
 printf(«Unable to send\n»);
 WSACleanup();
           system(«pause»);
 return 1;
 }
 char szResponse[256];
      //Пытаемся получить ответ от сервера
 retVal = recv(clientSock, szResponse, 256, 0);
 if(retVal == SOCKET_ERROR)
 {
```

```
 printf(«Unable to recv\n»);
 WSACleanup();
             system(«pause»);
 return 1;
 }
      char *Resp;
      Resp = szResponse;
      if(pBuf[0] != 's')
       printf(«a + b =  %s\n»,Resp);
      else
            printf(" %s\n»,Resp);
 closesocket(clientSock);
 WSACleanup();
 system(«pause»);
 return 0;
}
```

Приложение Б

ОСНОВНЫЕ ЕДИНИЦЫ ОБМЕНА ДЛЯ РАЗЛИЧНЫХ УРОВНЕЙ СТЕКА TCP/IP

Б.1. ТИПЫ КАДРОВ ТЕХНОЛОГИИ ETHERNET (РАЗМЕРЫ ПОЛЕЙ УКАЗАНЫ В БАЙТАХ)

Форматы четырех типов кадров Ethernet

Кадр 802.3/LLC

6	6	2	1	1	1(2)	46–1497 (1496)	4
DA	SA	L	DSAP	SSAP	Control	Data	FCS
			Заголовок LLC				

Кадр Raw 802.3/Novell 802.3

6	6	2	46–1500	4
DA	SA	L	Data	FCS

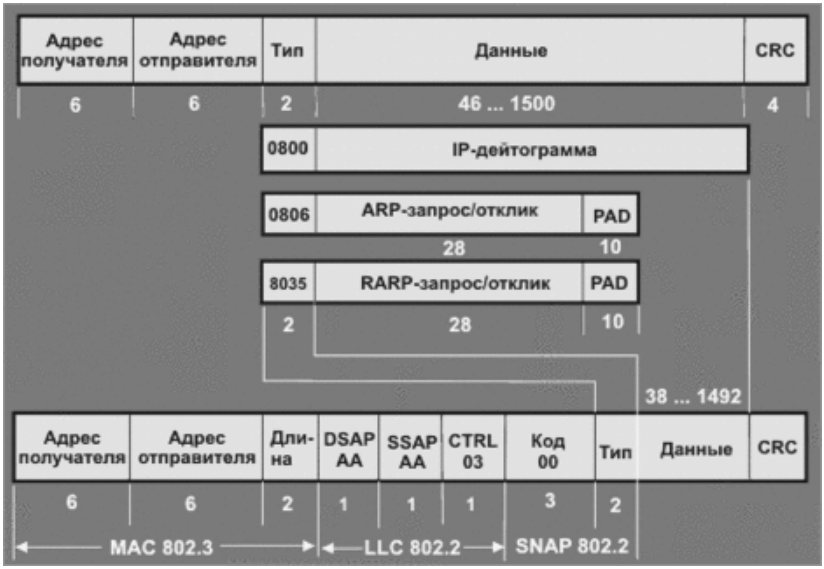
Кадр Ethernet DIX (II)

6	6	2	46–1500	4
DA	SA	T	Data	FCS

Кадр Ethernet SNAP

6	6	2	1	1	1	3	2	46–1492	4
DA	SA	L	DSAP	SSAP	Control	OUI	T	Data	FCS
			AA	AA	03	000000			
			Заголовок LLC			Заголовок SNAP			

Б.2. ETHERNET ИНКАПСУЛЯЦИЯ (RFC 894)

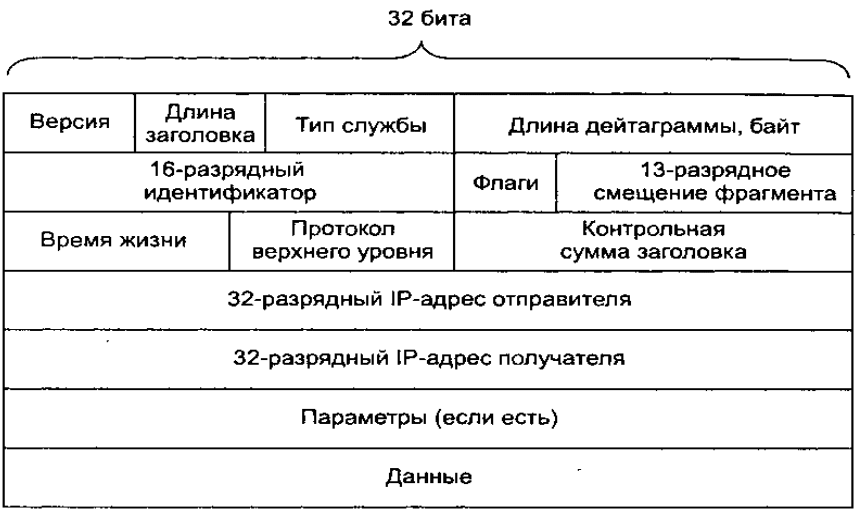


Размеры полей указаны в байтах

Б.3. АЛГОРИТМ ОПРЕДЕЛЕНИЯ ФОРМАТА КАДРА

Отличить один формат кадра Ethernet от другого не представляет большого труда, и сделать это можно с помощью следующего простого алгоритма. Сначала программа должна проверить значение поля типа протокола/длины кадра (13-й и 14-й байты в заголовке). Если записанное там значение превышает 0x05FE (максимально возможная длина кадра), то это кадр Ethernet_II. Если значение поля типа не превышает величины 1500, то следует продолжить проверку. Если первые два байта поля данных равны 0xFFFF, то это формат Ethernet_802.3 для NetWare 3.x. В противном случае это стандартный формат кадра 802.2, и остается только выяснить, какой из двух – обычный (Ethernet_802.2) или расширенный (Ether-net_SNAP). В случае Ethernet_SNAP значение первого, впрочем, как и второго, байта в поле данных равняется 0xAA. (Значение третьего байта равняется 0x03, но это проверять излишне.)

Б.4. КЛЮЧЕВЫЕ ПОЛЯ IPv4-ДЕЙТАГРАММЫ



- **Версия.** Четыре бита в этом поле определяют номер версии протокола IP. По этому номеру маршрутизатор может определить, как интерпретировать остальные поля IP-дейтаграммы. В различных версиях протокола IP применяются различные форматы IP-дейтаграмм. На рисунке показан формат дейтаграммы текущей версии протокола IP **(IPv4)**.
- **Длина заголовка.** Поскольку IPv4-дейтаграмма может содержать разное количество необязательных полей параметров (включаемых в заголовок IPv4-дейтаграммы), эти четыре бита необходимы для того, чтобы определить, где заканчивается заголовок и начинаются данные. В большинстве IP-дейтаграмм не содержатся поля параметров, поэтому обычно заголовок IP-дейтаграммы 20-байтовый.
- **Тип службы.** Поле типа службы (Type Of Service, TOS) было включено в заголовок IPv4-дейтаграммы, чтобы была возможность разделять IP-дейтаграммы на типы (например, выделять дейтаграммы, для которых требуется низкая задержка, или высокая пропускная способность, или высокая надежность). Так, может оказаться полезным отличать дейтаграммы реального времени (например? используемые в IP-телефонии) от прочего трафика (например FTP).

- **Длина дейтаграммы.** Это полная длина IP-дейтаграммы (*заголовок плюс данные*) в байтах. Поскольку размер этого поля 16 бит, теоретически максимальный размер IP-дейтаграммы может составлять 65 535 байт. Однако размер дейтаграмм редко превосходит 1500 байт и обычно ограничивается значением 576 байт.
- **Идентификатор, флаги, смещение фрагмента.** Эти три поля имеют отношение к так называемой IP-фрагментации. Этот вопрос мы подробно рассмотрим чуть позже. Интересно отметить, что новая версия протокола IP (IPv6) запрещает фрагментацию в маршрутизаторах.
- **Время жизни.** Поле времени жизни (Time To Live, TTL) позволяет гарантировать, что дейтаграммы не будут вечно циркулировать в сети (например, из-за существующей в течение долгого времени маршрутной петли). Значение этого поля уменьшается на единицу на каждом маршрутизаторе. Когда значение поля TTL достигает нуля, маршрутизатор отбрасывает дейтаграмму.
- **Протокол.** Это поле используется только тогда, когда IP-дейтаграмма достигает конечного адресата. Значение поля определяет протокол транспортного уровня, которому следует передать данные из IP-дейтаграммы. Например, значение 6 означает, что порция данных должна быть передана протоколу TCP, а значение 17 – протоколу UDP. Список всех возможных номеров имеется в RFC 1700, RFC 3232. Обратите внимание, что роль номера протокола в IP-дейтаграмме полностью аналогична роли номера порта в сегменте транспортного уровня. Номер протокола представляет собой «клей», связывающий вместе сетевой и транспортный уровни, тогда как номер порта связывает транспортный уровень с прикладным (в кадре канального уровня также есть специальное поле, связывающее канальный уровень с сетевым уровнем).
- **Контрольная сумма заголовка.** Контрольная сумма заголовка помогает маршрутизатору обнаруживать ошибки в полученных IP-дейтаграммах. Контрольная сумма заголовка вычисляется суммированием всех двухбайтовых слов заголовка в дополнительном коде. Маршрутизатор вычисляет контрольную сумму заголовка для каждой полученной дейтаграммы и таким образом проверяет ошибки в заголовке. Как правило, маршрутизаторы отбрасывают дейтаграммы, в которых обнаруживают ошибки. Обратите внимание, что контрольную сумму нужно вычислять заново и снова сохранять в поле заголовка на каждом маршрутизаторе, так как на единицу уменьшается поле времени жизни, могут также измениться поля параметров (описание быстрых алгорит-

мов для вычисления контрольной суммы заголовка IP-дейтаграммы содержится в RFC 1071).

- **IP-адреса отправителя и получателя.** Эти поля содержат 32-разрядные IP-адреса отправителя и конечного получателя IP-дейтаграммы.
- **Параметры.** Поле параметров позволяет расширить IP-заголовок. Параметры заголовка представляют собой редко используемые необязательные поля IP-дейтаграммы.
- **Данные (полезная нагрузка).** Наконец, мы добрались до последнего, самого важного поля, ради которого и существует дейтаграмма! В большинстве случаев поле данных IP-дейтаграммы содержит сегмент транспортного уровня (TCP или UDP), который необходимо доставить адресату. Однако поле данных может содержать и другие типы данных, например сообщения протокола ICMP.

Б.5. ФОРМАТ UDP-СООБЩЕНИЙ

Формат UDP-дейтаграммы

Основные поля

Длина UDP-сообщения равна числу байт в UDP-дейтаграмме, включая заголовок.

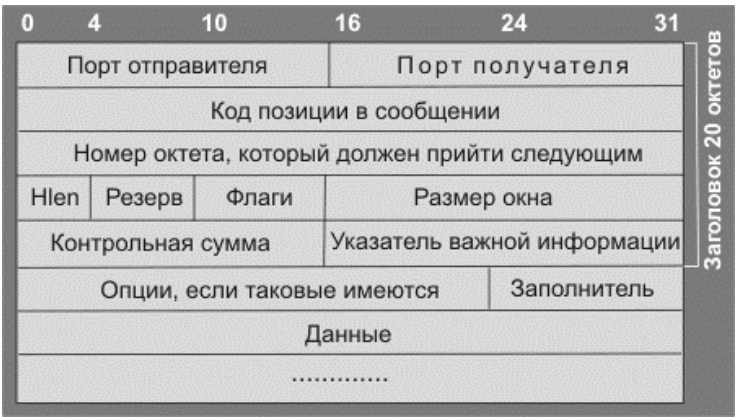
Контрольная сумма содержит код, полученный в результате контрольного суммирования UDP-заголовка и поля **данные**. (Нетрудно видеть, что этот протокол использует заголовок минимального размера (**8 байт**)).

Номера портов от 0 до 1023 стандартизованы – использовать их в прикладных задачах не рекомендуется, поэтому, прежде чем использовать какой-то порт в своей программе, следует заглянуть в RFC-1700.

Б.6. ФОРМАТ TCP-СЕГМЕНТА

Протокол **TCP (transmission control protocol**, RFC-793, -1323, -1644[T/TCP], -2018, -2581, -2582[RENO], -2861, -2873, -2883[SACK], -2923[MTU], -2988[RTO], -3293[GSMP], -3448[TFRC], -3465, -3481) в отличие от UDP осуществляет доставку дейтаграмм, называемых сегментами, в виде байтовых потоков с установлением соединения. Протокол TCP применяется в тех случаях, когда требуется гарантированная доставка сообщений. Он использует контрольные суммы пакетов для проверки их целостности и освобождает прикладные процессы от необходимости тайм-аутов и повторных передач для обеспечения надежности. Для отслеживания подтверждения доставки в TCP реализуется алгоритм «скользящего» окна. Внутренняя структура модуля TCP гораздо сложнее структуры UDP. Подобно UDP прикладные процессы взаимодействуют с модулем TCP через порты. Под байтовыми потоками здесь подразумевается то, что один примитив, например **read** или **write**, может вызвать посылку адресату последовательности сегментов, которые образуют некоторый блок данных (сообщение).

Хотя протоколы UDP и TCP могли бы для сходных задач использовать разные номера портов, обычно этого не происходит. Модули TCP и UDP выполняют функции мультиплексоров/демультиплексоров между прикладными процессами и IP-модулем. При поступлении пакета в модуль IP он будет передан в TCP- или UDP-модуль согласно коду, записанному в поле протокола данного IP-пакета.



Формат TCP сегмента

Если IP-протокол работает с адресами, то TCP, так же как и UDP, с портами. Именно с **номеров портов** отправителя и получателя начинается заголовок TCP-сегмента.

Поле **код позиции в сообщении** определяет порядковый номер первого октета в поле данных пользователя.

Поле **Hlen** – определяет длину заголовка сегмента, которая измеряется в 32-разрядных словах.

Далее следует поле **резерв**, предназначенное для будущего использования, в настоящее время оно должно обнуляться.

Поле **размер окна** сообщает, сколько октетов готов принять получатель. Окно имеет принципиальное значение, оно определяет число сегментов, которые могут быть посланы без получения подтверждения. Значение ширины окна может варьироваться во время сессии. Значение этого поля, равное нулю, также допустимо и указывает, что байты вплоть до указанного в поле *номер октета, который должен прийти следующим*, получены, но адресат временно не может принимать данные. Разрешение на посылку новой информации может быть дано с помощью посылки сегмента с тем же значением поля *номер октета, который должен прийти следующим*, но ненулевым значением поля ширины окна.

Поле **контрольная сумма** предназначено для обеспечения целостности сообщения.

Поле **указатель важной информации** представляет собой указатель последнего байта, содержащий информацию, которая требует немедленного реагирования.

Поле **опции** зарезервировано на будущее и в заголовке может отсутствовать, его размер переменен и дополняется до кратного 32 бит с помощью поля *заполнитель*.

Поле **данные** в TCP-сегменте может и отсутствовать, характер и формат передаваемой информации задаются исключительно прикладной программой, максимальный размер этого поля составляет в отсутствие опций 65 495 байт. TCP является протоколом, который ориентируется на согласованную работу хостов и программного обеспечения партнеров, участвующих в обмене информацией.

Долозов Николай Лаврентьевич

КОМПЬЮТЕРНЫЕ СЕТИ

Учебно-методическое пособие

Редактор *И.Л.Кескевич*
Выпускающий редактор *И.П. Бровановa*
Корректор *И.Е. Семенова*
Дизайн обложки *А.В. Ладыжская*
Компьютерная верстка *В.Н. Зенина*

Подписано в печать 24.12.2013. Формат 60×84 1/16. Бумага офсетная. Тираж 100 экз.
Уч.-изд. л. 6,51. Печ. л. 7,0. Изд. № 259. Заказ № 101. Цена договорная

Отпечатано в типографии
Новосибирского государственного технического университета
630073, г. Новосибирск, пр. К. Маркса, 20