Министерство образования и науки Российской Федерации

###### Новосибирский государственный технический университет

В.Г. КОБЫЛЯНСКИЙ

ЛОКАЛЬНЫЕ КОМПЬЮТЕРНЫЕ СЕТИ.

БАЗОВЫЙ КУРС

Новосибирск

2022

УДК 004.7:004.41

Рецензенты:

доктор техн. наук, профессор В.И. Хабаров

канд. физ.-мат. наук, доцент М.Г. Зайцев

**Кобылянский В. Г.**

Локальные компьютерные сети. Базовый курс: учеб. пособие /В. Г. Кобылянский. – Новосибирск: Изд-во НГТУ, 2022. – 118 с.

Учебное пособие по курсу «Операционные системы и компьютерные сети» предназначено для студентов, обучающихся по направлениям 01.03.02 «Прикладная математика и информатика», 02.03.01 «Математика и компьютерные науки»и 02.03.03 «Математическое обеспечение и администрирование информационных систем»и содержит описание общих понятийи принципов работы компьютерной сети, изложенных на основе эталонной модели OSI, а также описание основных принципов разработки сетевых приложений.

Пособие содержит примеры программ и может быть полезным для студентов всех направлений, желающихизучить основные принципы функционирования компьютерных сетей.

Работа подготовлена на кафедре «Теоретическая и прикладная информатика» и утверждена Редакционно – издательским советом университета в качестве учебного пособия

**УДК 004.7:004.41**

© Кобылянский В. Г., 2022

© Новосибирский государственный технический университет, 2022

Оглавление

[ВВЕДЕНИЕ 5](#_Toc118651290)

[1. ОСНОВНЫЕ ПОНЯТИЯ КОМПЬЮТЕРНЫХ СЕТЕЙ 7](#_Toc118651291)

[1.1 Классификации компьютерных сетей 7](#_Toc118651292)

[1.2 Способы коммутации 10](#_Toc118651293)

[1.3 Способы передачи данных 13](#_Toc118651294)

[1.4 Коммуникационное оборудование 14](#_Toc118651295)

[1.5 Топологии сетей 16](#_Toc118651296)

[1.6 Способы адресации сетевых узлов 19](#_Toc118651297)

[1.7 Физическая среда передачи данных 23](#_Toc118651298)

[1.8 Стандартизация сетевых взаимодействий 24](#_Toc118651299)

[1.8.1 Модель OSI 26](#_Toc118651300)

[1.8.2 Модель TCP/IP 31](#_Toc118651301)

[1.8.3 Модель IEEE 802 32](#_Toc118651302)

[1.9 Практическая работа 32](#_Toc118651303)

[1.9.1 Общие сведения 32](#_Toc118651304)

[1.9.2 Задание 33](#_Toc118651305)

[1.10 Контрольные вопросы 34](#_Toc118651306)

[2. ТЕХНОЛОГИИ КАНАЛЬНОГО УРОВНЯ 35](#_Toc118651307)

[2.1 Стандарты канального уровня 35](#_Toc118651308)

[2.1.1 Подуровень LLC 36](#_Toc118651309)

[2.1.2 Подуровень MAC 37](#_Toc118651310)

[2.1.3 Адресация канального уровня 38](#_Toc118651311)

[2.2 Основы технологии Ethernet 39](#_Toc118651312)

[2.2.1 Форматы кадров Ethernet 42](#_Toc118651313)

[2.3 Оборудование канального уровня 45](#_Toc118651314)

[2.4 Практическая работа 49](#_Toc118651315)

[2.5 Контрольные вопросы 50](#_Toc118651316)

[3. ТЕХНОЛОГИИ СЕТЕВОГО УРОВНЯ 51](#_Toc118651317)

[3.1 Адресация сетевого уровня 52](#_Toc118651318)

[3.2 Связь физических, сетевых и доменных адресов 56](#_Toc118651319)

[3.2.1 Протокол ARP 57](#_Toc118651320)

[3.2.2 Протокол DNS 59](#_Toc118651321)

[3.3 Протоколы сетевого уровня 60](#_Toc118651322)

[3.4 Основы IP-маршрутизации 65](#_Toc118651323)

[3.5 Диагностика IP-сети 70](#_Toc118651324)

[3.5.1 Команды диагностики IP-сетей 70](#_Toc118651325)

[3.5.2 Интерактивные средства диагностики IP-сетей 77](#_Toc118651326)

[3.6 Практическое задание 77](#_Toc118651327)

[3.7 Контрольные вопросы 78](#_Toc118651328)

[4. ТРАНСПОРТНЫЙ УРОВЕНЬ 80](#_Toc118651329)

[4.1 Протокол UDP 82](#_Toc118651330)

[4.2 Протокол TCP 83](#_Toc118651331)

[4.2.1 Установление и разрыв соединения 84](#_Toc118651332)

[4.2.2 Обмен данными 85](#_Toc118651333)

[4.3 Контрольные вопросы 88](#_Toc118651334)

[5. ОСНОВЫ РАЗРАБОТКИ СЕТЕВЫХ ПРИЛОЖЕНИЙ 90](#_Toc118651335)

[5.1 Общие сведения 90](#_Toc118651336)

[5.2 Функции для работы с сокетами 91](#_Toc118651337)

[5.2.1 Функция socket 91](#_Toc118651338)

[5.2.2 Функция bind 92](#_Toc118651339)

[5.2.3 Функция listen 93](#_Toc118651340)

[5.2.4 Функция accept 93](#_Toc118651341)

[5.2.5 Функция connect 93](#_Toc118651342)

[5.2.6 Функции send и sendto 94](#_Toc118651343)

[5.2.7 Функции recv и recvfrom 94](#_Toc118651344)

[5.2.8 Функция close 94](#_Toc118651345)

[5.3 Особенности разработки многопоточного сервера 95](#_Toc118651346)

[5.3.1 Общие сведения 95](#_Toc118651347)

[5.3.2 Управление потоками 97](#_Toc118651348)

[5.4 Практическое задание 98](#_Toc118651349)

[5.4.1 Разработка однопоточного сервера 98](#_Toc118651350)

[5.4.2 Разработка многопоточного сервера 100](#_Toc118651351)

[5.5 Контрольные вопросы 100](#_Toc118651352)

[БИБЛИОГРАФИЧЕСКИЙ СПИСОК 102](#_Toc118651353)

[КРАТКИЙ СЛОВАРЬ ТЕРМИНОВ 103](#_Toc118651354)

[ПРИЛОЖЕНИЕ 1 108](#_Toc118651355)

[ПРИЛОЖЕНИЕ 2. 112](#_Toc118651356)

ВВЕДЕНИЕ

Компьютерные сети — это одна из самых важных технологий нашего времени, обеспечившая переход на качественно новый уровень хранения, обработки и передачи данных.Сегодня невозможно представить, что еще каких-то 30 - 40 лет назад компьютер считался очень дорогим устройством, приобрести которое могли только большие научно-исследовательские организации, заводы и университеты, а работать с ним мог только специально обученный персонал. Тогда основным средством обмена данными между организациями была факсимильная связь, с помощью которой можно было передавать неподвижные изображения по телефоннымсетям общего пользования, причем время передачи одной страницы составляло 3-5 минут.

Создание компьютерных сетей привело не только к существенному увеличению скорости обмена данными между абонентами и появлению новых технологий, но изменило уклад жизни человечества. Социальные сети, насчитывающие сотни миллионов пользователей, становятся все более влиятельными инструментами влияния на образ жизни людей и даже на государственную политику. Современные информационные системы, образовательные платформы и системы обработки данных чаще всего реализуются с использованием удаленного доступа.

В настоящее время имеется устойчивый спрос на специалистов, предметной областью которых являются компьютерные сети. Перечень профессий таких специалистов очень широкий – сетевые инженеры, сетевые администраторы, разработчики сетевого программного обеспечения, разработчики сетевого оборудования и т.д.Специалисты других профессий IT-индустрии также должны владеть основными понятиями компьютерных сетей, знать состав основного сетевого оборудования и уметь настроить сетевую систему компьютера.

Современные сети, как правило, состоят из множества небольших локальных сетей, поэтому изучение желательно начинать именно с локальных сетей. Предлагаемое пособие содержит минимальный объем информации, необходимой для понимания основных принципов функционирования таких сетей.

Структура пособияпостроена на основе эталонной модели открытого взаимодействия систем OSI, являющейся фундаментом всей системысетевых стандартов и включает пять разделов.Первый раздел содержит описание общих понятий и основных принципов функционирования компьютерных сетей, последующие три раздела содержат описания технологий канального, сетевого и транспортного уровней, в пятом разделе излагаются основные принципы разработки сетевых приложений. В разделах имеются практические задания и контрольные вопросы, ответы на которые позволят студентам выделить ключевые моменты и закрепить полученные знания. Изучение материала желательно начать с первого раздела, остальные разделы можно изучать в произвольном порядке.

В пособии также приведены список рекомендуемых источников, словарь основных терминов и приложения, содержащие примеры реализации сетевых программ, написанных на языке C++.

Целевой аудиторией данного пособия являются молодые программисты, желающие понять основные принципы работы компьютерных сетей и особенности разработки прикладного программного обеспечения, реализующего удаленное взаимодействие.

1. ОСНОВНЫЕ ПОНЯТИЯ КОМПЬЮТЕРНЫХ СЕТЕЙ

*Прочитав эту главу, вы узнаете:*

* *классификации компьютерных сетей;*
* *способы коммутации и передачи данных;*
* *топологии сетей;*
* *способы адресации узлов;*
* *определение следующих терминов: информационно-телекоммуникационная сеть, коммуникационная сеть, протокол, интерфейс,инкапсуляция, многоуровневая модель сети*

## 1.1 Классификации компьютерных сетей

Компьютерная сеть – это совокупность однородных или разнородных цифровых устройств, объединенных между собой с помощью быстродействующих каналов связи. Такое объединение существенно расширяет возможности пользователей, делая для них доступнымитеоретически все ресурсы каждого устройства. Это позволяет организовать не только простой обмен данными между узлами сети, но и создавать мощные центры хранения и обработки данных, обладающие очень высокими производительностью.

Современные компьютерные сети являются очень сложными техническими объектами, включающими десятки миллионов узлов. Например, во время пандемии в марте 2020 года Microsoft сообщила о 40 млн ежедневных активных пользователях Skype и 75 млн ежедневных активных пользователях Team.

Наиболее общим видомсети является **информационно-телекоммуникацион-наясеть** (ИТС)**.** Государственный стандарт ГОСТ Р 52653-2006 определяет это понятие, как технологическую систему,предназначеннуюдляпередачиполиниямсвязиинформации,доступккоторойосуществляетсясиспользованиемсредстввычислительнойтехники. Наиболее известным примером ИТС является **глобальная сеть** «Интернет», включающая сотни миллионов устройств.

Обобщенная модель ИТС имеет трехуровневую иерархическую структуру и приведенана рис.1.1.

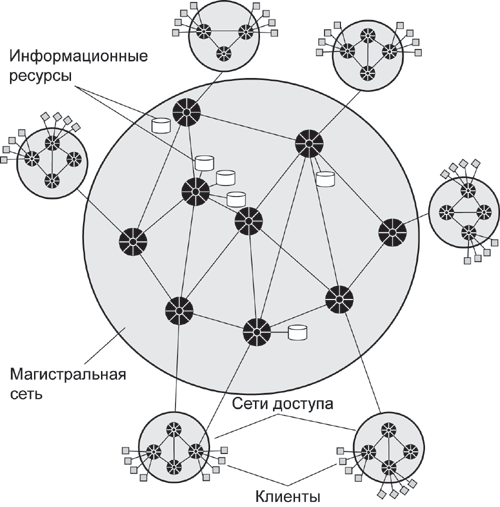


Рис.1.1

На нижнем уровне находятся терминальные устройства (ТУ) пользователей, являющиесяпоставщиками или потребителями информации. Основными ТУ в компьютерных сетях являются компьютеры, в телевизионных сетях – телевизоры, в телефонных сетях – телефоны. В компьютерных сетях в качестве ТУ могут также использоваться другиецифровые устройства – кассовые аппараты, банкоматы, смартфоны, видеокамеры, часы, бытовая техника. Информация от ТУ по каналам связи поступает на коммутаторы сетей доступа.

**Сеть доступа** представляет средний уровень ИТС и, в свою очередь, также может иметь несколько уровней,например, провайдер-университет-факультет. Нижний уровень такой сети объединяет компьютеры, находящиеся на относительно небольшой территории и называется **локальной сетью.**

Основные функции сети доступа:

* мультиплексирование – объединение потоков информации, поступающей с компьютеровпользователей в один агрегированный поток и передача этого потока в коммутатор магистральной сети,
* демультиплексирование – прием агрегированного потока информации, разделение его на отдельные потоки и передача потоков на компьютерыпользователей.

**Магистральная сеть** представляет верхний уровень ИТС и предназначена для передачи агрегированных потоков из сети доступа отправителей в сеть доступа получателей.

**Информационные центры**- это собственные информационные ресурсы сети, на основе которых осуществляется обслуживание пользователей. В таких центрах может храниться информация двух типов:

* пользовательская информация (например, Web-порталы, на которых хранятся справочная информация, новости, каталоги электронных магазинов и т.п.);
* служебная информация, обеспечивающая функционирование ИТС (например, система DNS-серверов) или позволяющая предоставлять пользователям определенные услуги (например, системыхранения учетной информации пользователей; системы аутентификации и авторизации пользователей, используемые для проверки прав пользователей на получение услуг; системы биллинга, подсчитывающие плату за предоставленные услуги).

Основными компонентами ИТС являются сетевое оборудование и сетевое программное обеспечение (ПО). Оборудование включает два компонента – коммуникационные устройства (сетевые карты, коммутаторы, маршрутизаторы, шлюзы, мультиплексоры и т.д.) и линии связи, которые могут быть кабельными, оптическими или беспроводными. Для подключения к линиям связи все коммуникационные устройства имеют набор портов.

Сетевое оборудование имеет очень большую номенклатуру и разрабатывается и производится сотнями разных производителей. Поэтому для обеспечения высокой надежности взаимодействия между собой разнородного оборудования разработано более 700 протоколов, которые реализуютсяспециальным сетевым ПО.Протокол определяет формат и очередность сообщений, которыми обмениваются два или более устройства, а также действия, выполняемые при передаче и/или приеме сообщений или при наступлении определенных событий.

Локальная сеть является частным случаем ИТС и обычно может быть представлена в упрощенном виде, показанном на рис. 1.2. Здесь **коммуникационная сеть**– это совокупность коммуникационногооборудования и линий связи, предназначенных для организации обмена информацией, представленной в требуемой форме.

Коммуникационнаясеть

Шлюз

Рис.1.2

Основными свойствами компьютерных сетей являются способ коммутации, топология и пропускная способность, определяющая максимальную скорость передачи данных и измеряемая количеством битов, передаваемых в секунду (бит/с).

*Обратите внимание: более крупные единицы производительности (кбит/с, мбит/с, гбит/с) связаны с основной единицейкоэффициентом 1000. Таким образом, в отличие от единиц измерения объема информации, 1кбит/с = 1000 бит/с, 1 мбит/с = 1000 кбит/с и т.д.*

## 1.2Способы коммутации

Коммутация – процесс взаимодействия устройств сети для продвижения данных от отправителя адресату через транзитные узлы. Известнытриспособа коммутации: коммутация каналов, коммутация пакетов и организация виртуального канала.

В сетях с **коммутацией каналов** (КК)предполагается создание временного соединения (канала) для организации обмена данными между двумя узлами сети с организацией соединения типа “точка-точка”. При этом поток данных заранее получает маршрут, задаваемый администратором или рассчитанный динамически. Каналв течение всего сеанса обмена информацией может использоваться только указанными двумя узлами. После завершения обмена соединение разрывается и линии связи считается свободными.

В настоящее время этот способ используется в телефонных сетях, а также для построения высокоскоростных магистральных каналов. Недостаток – неэффективное использование канала при сильно пульсирующем трафике, которым характеризуются компьютерные сети.

Коммутация каналов основана на понятии **элементарного канала** - базовой технической характеристики, определяющей фиксированное значение ***пропускной способности***. Любая линия в сетях КК имеет пропускную способность, кратную элементарному каналу, который равен 64 кбит/с. Такая скорость достаточна для качественной передачи голоса с частотой дискретизации 8000 Гц. При дискретизации с этой частотой и кодировании каждого кванта звука одним байтом (256 градаций) получаем 8000 \* 8 = 64 кбит/с.

На рис.1.3 приведен пример организации составного канала для соединения абонентов А и B через транзитные узлы S1-S4. Канал содержит несколько линийсвязи, содержащих от 2 до 5 элементарных каналов. Пропускная способность составного канала определяется самой медленной линией связи из всех, включенных с канал.Поэтому максимальная скорость обмена данными между абонентами AиB в приведенном примере будет равна 128 кбит/с.

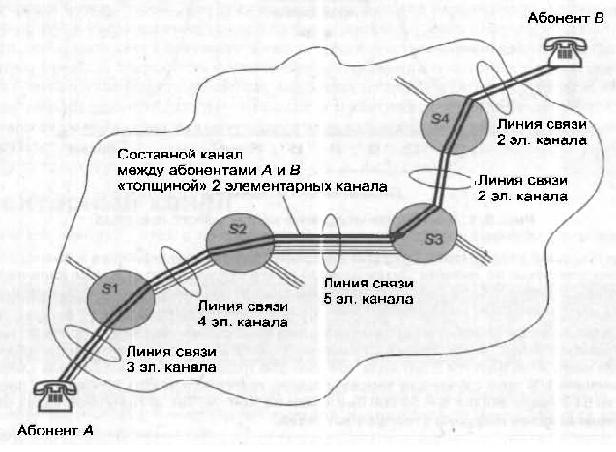


Рис. 1.3

Составной канал имеет следующие свойства:

* имеет фиксированный маршрути фиксированнуюпропускную способность;
* все элементарные каналы, входящие в составной канал, на время сеанса поступают в исключительное пользование двух абонентов, а после окончания сеанса освобождаются;
* абоненты могут посылать в сеть данные со скоростью, не превышающей пропускную способность составного канала;
* данные гарантированно поступают адресату без задержек и потерь.

Сеанс передачи происходит в два этапа: а) в сеть поступает служебное сообщение – запрос, который содержит адрес абонента и организует составной канал; б) по подготовленному составному каналу передается поток данных без вспомогательной информации (например, без адреса абонента).

Сеть КК эффективно передает трафик, если его скорость постоянна в течение всего сеанса связи и совпадает с пропускной способностью канала. При передаче голоса, когда идет двусторонний обмен информацией, полоса пропускания используется эффективно, но компьютерный трафик имеет пульсирующий или даже взрывной характер, если канал загружают разные приложения. Поэтому для оптимального использования ресурсов в сетях с неоднородным трафиком был разработан способ коммутации пакетов.

В сетях с **коммутацией пакетов**(КП)сетевое оборудование разбиваетпередаваемые сообщения на небольшие блоки (пакеты**),** которые передаются отдельно друг от друга. Каждый пакет снабжается заголовком, в котором содержатся номер пакета, адрес назначения и другая вспомогательная информация. Узел-приёмник собирает сообщение из полученных пакетов.

В сетях КП по одной физической линии связи могут обмениваться данными несколько узлов. Пакеты могут иметь постоянную или переменную длину в зависимости от технологии. Недостаток такого способа – усложнение сетевого оборудования.

На рис. 1.4 показана передача пакетов в сети, содержащей три узла (3,4,10) и два коммутатора (5,8). Из рисунка видно, что коммутатор 5 передает все пакеты с узлов 3,4,10 в коммутатор 8 в режиме мультиплексирования.

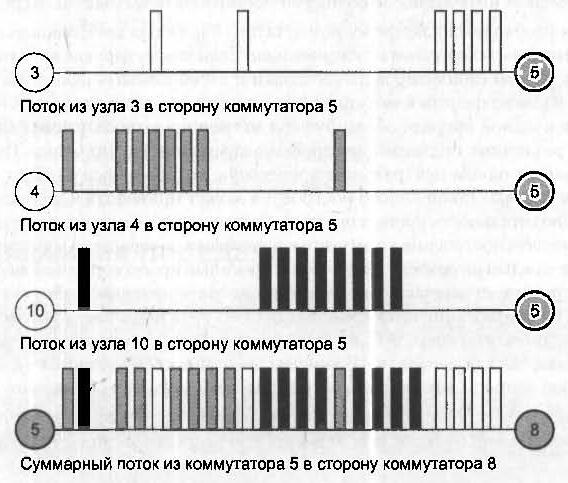


Рис. 1.4

В основу сети КП положены следующие базовые принципы:

* отправитель и получатель должны координировать передачу данных для обеспечения их надёжной доставки;
* сетевое оборудование должно обеспечить равноправный бесперебойный доступ компьютеров к разделяемым линиям связи;
* пакетный коммутатор должен иметь внутреннюю память для временного хранения пакетов, чтобы согласовать скорость приема полученных пакетови скорость их обработки.

Современные компьютерные сети в основном используют способ коммутации пакетов.

## 1.3Способы передачи данных

В сетях КП возможны три способа передачи данных: дейтаграммный, с предварительным установлением соединения, с установлением виртуального канала.

**Дейтаграммный** способ основан на том, что каждый пакет рассматривается как независимая единица (дейтаграмма), все пакеты продвигаются независимо друг от друга на основании одних и тех же правил. Никакая информация об уже переданных сетью пакетах не хранится. Способ работает быстро, но не гарантирует доставку пакета адресату, он делает это по возможности.

**Предварительное установление логического соединения** предполагает согласование двумя конечными узлами некоторых параметров обмена пакетами, эта процедура называется установлением логического соединения. Способ продвижения пакетов при этом основан на знании устройствами сети истории обмена данными, например, узел-отправитель запоминает число отправленных пакетов, а получатель – число принятых.

Наличие логического соединения позволяет более рационально обрабатывать пакеты, например, при потере нескольких пакетов можно уменьшить скорость передачи. Можно также увеличить надежность за счет отбрасывания дубликатов или повторной передачи потерянных пакетов. Этот способ медленнее дейтаграммного и требует передачи двух типов пакетов – информационных и служебных.

**Виртуальный канал** является комбинацией двух предыдущих способов. В этом случае в сети с коммутацией пакетов организуется заранее проложенный фиксированный маршрут, соединяющий конечные узлы. Такие каналы прокладываются для *устойчивых* информационных потоков, каждый пакет такого потока помечается специальной **меткой**.

Сеанс начинается с запроса на установление соединения с помощью служебного пакета, в котором указывается адрес назначения и метка потока, для которого прокладывается виртуальный канал. Запрос, проходя по сети, в каждом коммута-торе создает отдельную запись, указывающую, как коммутатор должен обрабаты-вать пакеты с заданной меткой. После установления соединения проводится передача данных, причем во всех пакетах вместо адреса назначения указывается метка виртуального канала.

## 1.4Коммуникационное оборудование

Основными компонентами сети являются ТУ, передающая среда (проводная или беспроводная) и коммуникационное оборудование. Основным типом ТУ в компьютерных сетях являются компьютеры, на которых пользователи сети реализуют прикладные задачи, и серверы**,**предназначенные для хранения больших объемов данных и управления распределением сетевых ресурсов общего доступа. Сетевые компьютеры, подключенные к сети, часто называют рабочими станциями или хостами.

Сетевое коммуникационное оборудованиеимеет большую номенклатуру и может включать не только аппаратные средства, но и программное обеспечение, реализующее сложные алгоритмы обработки передаваемых данных. В последующих разделах пособия эти алгоритмы будут рассмотрены более подробно. На данном этапе рассмотрим только основные типы сетевого оборудования и их назначение.

**Сетевая карта (сетевой адаптер, сетевой интерфейс)** –контроллер, расположенный на материнской плате компьютера и предназначенный для приема сигналов из сети и передачи сигналов в сеть.

**Повторитель** (Repeater) — устройство, имеющее два порта и предназначенное для усиления и восстановления формы сигналов, полученных на входной порт, и передачи его на выходной порт.

**Концентратор** (Hub) –устройство, имеющее несколько портов. При получении сигнала на любой порт концентратордублирует его на все остальные порты, являясь по сути многопортовым повторителем. Основными отличиями концентратора от повторителя являются возможность подключения портов к сетям с разной передающей средой (например, коаксиальный кабель и витая пара) и автоматическое отключение портов при возникновении на них ошибок.

**Мост** (Bridge) – устройство, имеющее встроенный микропроцессор и несколько портов. При получении сигналов на любой порт микропроцессор проводит анализ входных данных, выделяя из них блоки, которые называются кадрами. Из полученного кадра определяется адрес получателя и далее этот кадр передается только на тот порт, к которому подключен получатель. Обработка данных, получаемых на разные порты, микропроцессором проводится последовательно. Мост также усиливает сигналы и при необходимости конвертирует сигналы для кабеля другого типа.

**Коммутатор** (Switch) –это мост, у которого каждый порт управляется собственным микропроцессором, за счет чего резко увеличивается производительность устройства. Коммутатор имеет встроенную память и может одновременно обрабатывать несколько кадров. Если по какой-то причине нужный порт в данный момент времени занят, то кадр помещается в память и ожидает своей очереди. Построенные с помощью коммутаторов сети могут охватывать большое число компьютеров и иметь протяженность в несколько километров.

**Маршрутизатор** (Router) – многопортовое устройство, каждый порт которого подключается к отдельной сети. Маршрутизатор используется для организации передачи данных между сетями, выполняя функцииформирования маршрута от отправителя к получателю и фильтрации широковещательных сообщений.Маршрутизация проводится на основе анализа сетевого адреса получателя, который содержится в каждом принятом пакете данных.

**Мультиплексор** – это устройство, поддерживают несколько сотен цифровых абонентских линий. Мультиплексоры посылают и получают абонентские данные по низкоскоростным линиям связи, концентрируя весь трафик в одном высокоскоростном канале для передачи в Интернет или в сеть компании.

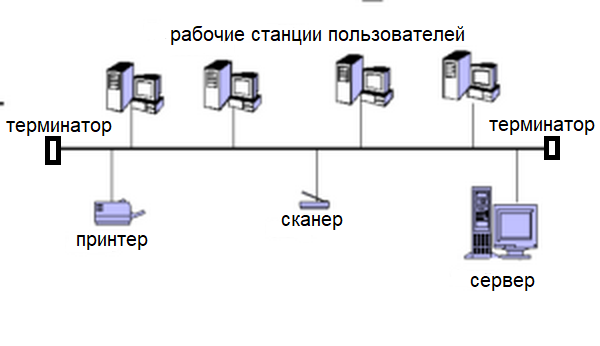
**Межсетевой экран** (firewall, брандмауэр) - это устройство, реализующее контроль за поступающей в локальную сеть и исходящей из нее информацией и обеспечивающие защиту локальной сети посредством фильтрации трафика.

**Шлюз** (Gateway) - многопортовое устройство,соединяющие разнородные сети или сетевые устройства (например, проводные и беспроводные). Шлюзы позволяет решать проблемы различия протоколов и систем адресации. Обычно шлюз является узлом сети, через который проводится выход в другую сеть. Например, корпоративная сеть обычно подключается к сети провайдера через шлюз, выполняющий функции межсетевого экрана.

## 1.5Топологии сетей

Топология сети – конфигурация графа, вершинам которого соответствуют ТУ и коммутационное оборудование, а ребрам – физические или информационные связи между вершинами.

Базовыми топологиями сетей являются шина, кольцо и звезда. На рис.1.5а показана структура сети с **общей шиной**. Топология с общей шиной применяется, если в сеть соединяются близко расположенные узлы, например, в соседних помещениях. Шина проводит сигналы из одного конца сети к другому. На концах шины устанавливаются специальные резисторы-терминаторы для того, чтобы на концах шины не возникало искажений сигналов из-за их отражения.



а) б)

Рис.1.5

Все рабочие станции подключаются к единой шине, поэтому при передаче данных одной станцией все другие станции принимают эти сигналы. Каждая рабочая станциявыделяет из полученных данных адрес получателя, и если он совпадает с адресом рабочей станции, то она его принимает. Если одна из станций не работает, то это не сказывается на работе сети в целом.

Для того, чтобы исключить одновременную посылку данных несколькими узлами сети применяется контроль «несущего» сигналаили один из узловсчитается главным, управляя очередностью передачи. Например, в сетях Ethernet c шинной топологией (IEEE 802.3) рабочие станции всегда передают свои данные только в случае отсутствия несущей частоты в среде передачи данных, что говорит о том, что среда свободна.

По топологии “шина” строились ранние локальные сети на коаксиальном кабеле. Шина прокладывалась через все помещения и подходила к каждойрабочей станции. При этом в качестве шины выступали отрезки коаксиального кабеля, соединенные Т-коннекторами. Боковой вывод Т-коннектора вставлялся в разъем на сетевой карте, в крайних рабочих станциях к Т-коннектору вместо кабеля подключался терминатор (рис. 1.5б).

В современных сетях логическая топология “шина” применяется в беспроводных сетях, так как все рабочие станции таких сетей используют общую разделяемую среду передачи данных и принимают сигналы всех соседних станций.

Топология **«кольцо»** представляет собой последовательное соединение компьютеров, когда последний соединён с первым (рис. 1.6). Сигнал проходит по кольцу от компьютера к компьютеру в одном направлении до тех пор, пока не дойдет до адресата.При этом каждый компьютер работает как повторитель, усиливая сигнал и передавая его дальше.

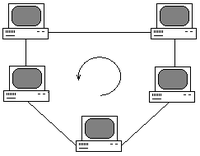


Рис. 1.6

В кольцевой топологии невозможно возникновение конфликтов между рабочими станциями, так как здесь определена жесткая очередность передачи данных для каждой станции. Поэтому кольцевая топология характеризуется более высокой надежностью сети в сравнении с общей шиной.

На рис. 1.7 приведены структура сети с топологией “звезда”. Здесь роль центрального блока могут выполнять коммуникационные устройства (концентратор, коммутатор) или отдельный управляющий компьютер (сервер). Эта топология часто используется при централизованной обработке данных или когда компьютеры разнесены территориально.

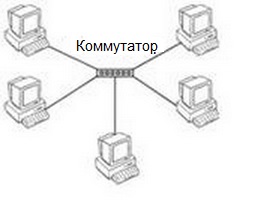


Рис. 1.7

Звезда является наиболее распространённой топологией, так как проста в обслуживании. В ней в качестве физической среды передачи данных используются витая пара или оптоволоконный кабель.

В таблице 1.1 приведены достоинства и недостатки каждой из базовых топологий.

Таблица 1.1

|  |  |  |
| --- | --- | --- |
| Топология | Достоинства | Недостатки |
| Шина | 1) Отказ любой из рабочих станций не влияет на работу всей сети.  2) Простота подключения новых узлов.  3) Невысокая стоимость реализации в связи с сокращением расхода кабеля. | 1) Разрыв кабеля, или другие неполадки в соединении может исключить нормальную работу всей сети.  2) Ограниченная длина кабеля и количество рабочих станций.  3) Трудно обнаружить дефекты соединений.  4) Невысокая производительность.  5) При большом объеме передаваемых данных главный кабель может не справляться с потоком информации, что приводит к задержкам. |
| Кольцо | 1) Резервирование связей за счет использования двух путей передачи данных– по часовой стрелке и против нее.  2) Возможность организации обратной связи (источник может проконтролировать доставку данных адресату).  3) Высокая надежность из-за отсутствия конфликтов между узлами. | 1) При выходе из строя одного узла нарушается работа сети. |
| Звезда | 1) Подключение новых рабочих станций не вызывает особых затруднений.  2) Возможность мониторинга сети и централизованного управления сетью.  3) При использовании централизованного управления сетью локализация дефектов соединений максимально упрощается.  4)Хорошая возможность масштабирования и модернизации. | 1) Отказ центрального узла приводит к отключению от сети всех рабочих станций, подключенных к ней.  2) Достаточно высокая стоимость реализации в связи с большимрасходом кабеля.  3) Число узлов ограничивается числом портов центрального узла. |

На основе базовых топологий возможнопостроение более сложных топологий– иерархической(многоуровневая звезда) и гибридной (смешанной). В этом случае в качестве узлов могут выступатьне только отдельные ТУ, но и локальные сети, использующие различные базовые топологии.Например, иерархичскаядревовидная топология корпоративной сети может повторять структуру организации, когда к главномусерверу,расположенному в центральном офисе, подключаются серверы подразделений, каждый из которых подключен к локальной сети, узлами которой являются компьютеры сотрудников подразделения.

Для сокращения трафика сеть может разбиваться на отдельные сегменты, соединенные между собой через коммуникационные устройства (коммутаторы или маршрутизаторы). Под сегментом понимается физически или логически обособленная часть сети. Коммутаторы работают с физическими сегментами, маршрутизаторы – с логическими сегментами (подсетями).

Сетевая топология может быть:

* физической, котораяпоказывает реальное расположение узлов сети и связей между ними на уровне монтажной документации (места установки оборудования и укладки кабелей, какой кабель в какой порт подключен, длина кабельных пролетов и т.д.);
* логической, которая показывает прохождение данных в рамках физической топологии;;
* информационной, которая показывает направление потоков информации, передаваемых по сети;
* управления обменом, которая отображает принцип и последовательность передачи права на захват сети отдельными узлами сети.

Одна сеть может описываться разными топологиями. Например, сеть с физической топологией «Звезда» с центральным узлом в виде концентратора имеет логическую топологию «Шина», т.к. в такой сети данные, передаваемые любым узлом, будут приниматься всеми другими узлами.

## 1.6Способы адресации сетевых узлов

Каждый узел сети должен иметь собственный уникальный адрес, по которому к нему могут обращаться другие узлы. Способы адресации зависят от технологии, на базе которой строится та или иная сеть.

Современные глобальные сети состоят из множества иерархически организованных подсетей, каждая из которых может использовать собственную технологию. Например, технологии X.25 и Frame relay, которые широко использовались в больших корпоративных сетях передачи данных в конце 90-х годов, и современное семейство технологийEthernet и Интернет предполагают различные способы адресации.

Любой узел должен иметь два адреса –локальный и сетевой. **Локальный (физический)**адрес узла позволяет идентифицировать узел только в пределах локальной сети и определяется технологией, на основе которой построена эта локальная сеть. Большинство современных технологий (Ethernet, Wi-Fi, Token Ring) для однозначной адресации используют MAC-адрес сетевой карты или порта коммуникационного устройства, имеющий длину 48 бит (6 байт).MAC-адрес записывается в виде последовательности чисел, каждое из которых является шестнадцатиричным значением соответствующего байта адреса, например, 76:12:B3:29:E3:ED.‬

MAC -адреса обычно назначаются производителем оборудования и являются уникальными адресами, т.к. распределяются централизованно.При этом первые 3 байта содержат код производителя, а следующие 3 байта являются уникальным номером устройства. ‬

**Сетевой адрес** используется для доступа к узлу из внешних сетей и не должен зависеть от способа адресации в локальных сетях.Поэтому сетевой адрес обычно содержитномер сети и номер узла в пределах сети.Форма записи сетевого адреса зависит от используемого стека протоколов и для наиболее распространенного в сети Интернет стека протоколов TCP/IP называется IP-адресом.IP-адрес занимает 4 байта, назначается администратором сети во время конфигурирования компьютеров и маршрутизаторов изаписывается в виде последовательности чисел, каждое из которых является десятичным значением соответствующего байта адреса, например, 217.71.131.243.

Одно сетевое устройство может входить в несколько сетей и иметь несколько IP-адресов. Например, каждый порт маршрутизатора по определению обслуживает свою сеть и должен иметь собственный IP-адрес, или в один компьютер можно установить несколько сетевых карт, каждая из которых имеет собственный IP-адрес. Таким образом IP-адрес характеризует один **сетевой интерфейс**, а не отдельный компьютер или маршрутизатор.

Сетевой адрес не связан с MAC-адресом. Определение MAC-адреса по известному IP-адресу проводится с помощью протокола ARP. Для этого на каждом сетевом интерфейсе хранится специальная ARP-таблица соответствия этих адресов.

Сетевые адреса могут быть нескольких типов: внешние (белые), локальные (серые) и специальные. **Внешние IP-адреса**доступны из сети Интернет, назначаются провайдерами и являются уникальными.Провайдеры услуг Интернет получают диапазоны адресов у подразделений специальной управляющей организации Интернет NIC (Network Information Center), а затем распределяют их между своими клиентами.

**Локальные IP-адреса** предназначены для обеспечения внутренней связности сети, что означает обеспечение доступа только внутри корпоративной сети организации. В этом случае уникальность адресов может обеспечивать администратор сети, причем в разных автономных сетях адреса могут повторяться. Для избежания возможных конфликтов с централизованно назначенными адресами стандартами Интернета предусмотрены определенные диапазоны локальных адресов, рекомендуемые для использования в локальных сетях: 10.0.0.0 -10.255.255.255, 172.16.0.0 -172.31.255.255, 192.168.0.0 -192.168.255.255.

Множество организаций используют внутри своих сетей одно и то же множество локальных адресов, что позволяет экономить внешниеIP-адреса. Для обеспечения подключения локальной сети к сети Интернет достаточно одного маршрутизатора, имеющего общий внешний адрес.

Сетевые адреса в виде IP-адреса трудно запомнить, поэтому для удобства пользователей была введена другая форма представления сетевого адреса в виде **доменного (символьного)** имени. Система доменных имен строится по иерархическому принципу и позволяет однозначно идентифицироватьузел, подключенный к сети Интернет.Официальный список всех доменов верхнего уровня поддерживается администрацией адресного пространства Интернет (IANA).

В качестве домена верхнего уровня используется код страны (для России - «ru», для Франции – «fr» и т.д.) или вид деятельности организации (gov – правительственные, edu – образовательные, com – коммерческие, org – общественные, net – организации, предоставляющие сетевые услуги). В пределах составной сети имена строятся по иерархическому принципу (рис. 1.8):

*компьютер.организация.город.страна*

**fpm2.ami.nstu.ru**

Уровень 4

Уровень 3

Уровень 2

Уровень1

Рис. 1.8

Например, доменное имя fpm2.ami.nstu.ru принадлежит серверу с именем fpm2 факультета прикладной математики и информатики (ami) Новосибирского государственного технического университета (nstu), расположенного в РФ (ru). Доменное имя назначается администратором соответствующего домена.

Между доменным именем и IP- адресом нет однозначной связи и единственный способом установления соответствия является использование таблиц соответствия. Для этого в сетях TCP/IP используется служба доменных имен (DNS), которая предназначена для установления соответствия между доменным именем и IP- адресом. Поэтому доменные имена часто называются DNS-именами.

**Специальные адреса** предназначены для широковещательных и групповых рассылок, а также для отладки сетевых приложений. Например, адреса, в которых первый байт равен 127, являются адресами внутреннего стека протоколов TCP/IP компьютера или маршрутизатора и могут называться терминами «внутренняя петля» или «loopback».Они используются для тестирования сетевого программного обеспечения или для организации клиент-серверного взаимодействия на одном компьютере.

Для определенияадресов узлов сети и маршрутов передачи данных междуними необходимо использовать сетевые команды ОС, представленные в таблице 1.2

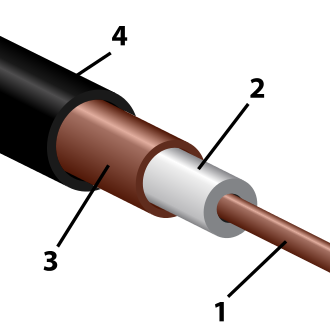
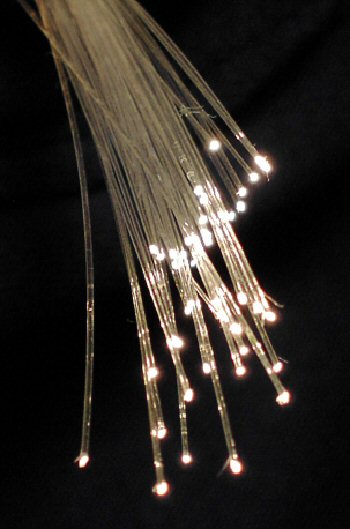
Таблица 1.2

|  |  |  |  |
| --- | --- | --- | --- |
| ОС | Команда | Пример | Назначение |
| Linux | ifconfig | ifconfig | Вывод характеристик сетевых интерфейсов текущего узла, включая IP- и MAC-адреса |
| traceroute имя\_узла | traceroute 192.168.130.5 | Вывод маршрута передачи пакетов от текущего узла сети к другому узлу. Для каждого промежуточного узла выводятся его символьное имя и IP-адрес. |
| Windows | ipconfig | ipconfig /all | Аналогично команде ifconfig |
| tracert имя\_узла | tracert 217.71.130.131 | Аналогично команде traceroute |

В последующих разделах пособия вопросы, связанные с адресацией, будут рассмотрены более подробно.

## 1.7Физическая среда передачи данных

Физические среды можно разделить на два типа: проводные и беспроводные. Проводные среды передачи предполагают наличие твердотельного проводника и включают коаксиальный кабель,медную витую пару и оптоволоконный кабель (рис.1.8). В беспроводной среде передача осуществляется с применением радиоканалови используется в беспроводных локальных сетях и в спутниковой связи.

****

а) б) в)

Рис.1.9

**Коаксиальный кабель**(рис.1.9а)состоит из двух медных проводников,расположенных вдоль оси кабеля (центральная жила и экранирующая оплетка).Такой кабель использовался в качестве разделяемой среды передачи данных в ранних версиях технологии Ethernet (до 10 Мбит/с). В настоящее время в компьютерных сетях практически не используется и применяется в основном в телевизионных сетях.

**Витая пара**(рис. 1.9б) является самым недорогим и наиболее популярным видом кабелей. Она представляет собой несколько пар изолированных проводников, скрученных между собой и покрытых пластиковой оболочкой. Скрутка позволяет снизить влияние электрических помех на передаваемые данные.

Качество кабеля с витой парой определяется его категорией, обозначаемых целыми числами от 1 до 8, некоторые категории делятся на подкатегории, например, 5е или 6а. В кабелях выше шестой категорий все пары проводов или каждая пара в отдельности дополнительно экранируются оплетками из фольги для увеличения помехозащищенности. Кабели категории 8 могут передавать данные со скоростью до 40 Гбит/с.

В настоящее время для построения компьютерных сетей в основном применяется кабель категории 5е, которая может обеспечить передачу данных со скоростью до 1Гбит/с.

**Оптоволоконныйкабель** (рис. 1.9в) представляет собой тонкий и гибкий кабель, внутри которого находится множество световых волокон (нитей), по которым передаются световые импульсы, несущие информацию о передаваемых битах. Достоинством такого кабеля является высокая пропускная способность (до 100 Гбит/с), высокая помехозащищенность, возможность передачи данных на большие расстояния (десятки километров без дополнительных усилителей, тысячи километров – с усилителями). Недостатком можно считать более высокую стоимость в сравнении с витой парой.

**Радиоканал**использует электромагнитноеизлучение радиодиапазона. Его основное достоинство – отсутствие кабельной системы, что дает возможность подключения к сети подвижных объектов. Характеристики радиоканала сильно зависят от состояния среды передачи радиоволни расстояния между оконечными системами. Состояние среды передачи определяется затуханием сигнала вследствие прохождения через поглощающие предметы и взаимодействием с отраженными электромагнитными волнами, а также волнами, исходящими от других источников излучения.

Спутники связи организуют взаимодействие между двумя или более наземными станциями. Они принимают сигналы одного частотного диапазона, проводят их обработку и усиление, а затем передают сигналы в другом частотном диапазоне. Скорость обмена данными, обеспечиваемая спутниковыми каналами, составляет несколько гигабит в секунду. Существуют два типа спутников: геостационарные и низкоорбитальные.

## 1.8Стандартизация сетевых взаимодействий

Классическим способом анализа сложных технических системявляется их декомпозиция – разбиение на отдельные подсистемы. Это дает возможность упростить задачу, представляя каждую подсистему в виде черного ящика, реализующего определенные функции и взаимодействующего с соседними подсистемами через определенные интерфейсы.Внутренняя реализация черного ящика в любой момент может быть изменена, не затрагивая соседние подсистемы.

Компьютерные сети не являются исключением и поэтомуосновой стандартизации здесь являются многоуровневые модели, в которых каждый уровень представляет отдельную независимую подсистему.Особенностью сетей,в сравнении с другими сложными объектами,является то, что в обмене данными принимают участие два удаленных друг от друга узла.В таком случае каждый уровень должен взаимодействовать как с выше- и нижележащими уровнями своего узла, так и с одноименным уровнем другого узла (рис. 1.10).

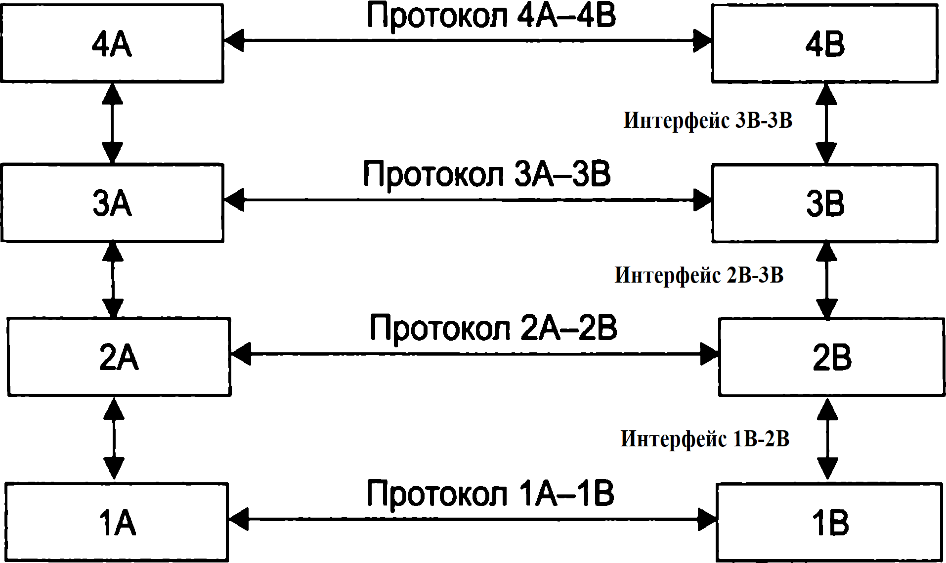


Рис. 1.10

Компьютерные сети объединяют большую номенклатуру цифровых устройств, выпускаемых различными производителями, причем часто производители разрабатывают собственное программное обеспечение для управления этими устройствами. Поэтому одной из наиболее острых проблем в обеспечении надежной работы сети является совместимость оборудования, которая обеспечивается наличием определенных стандартов, определяющих порядок взаимодействия разнородных устройств.

Совокупность соглашений, используемых при обмене данными между различными уровнями одного узла, называется интерфейсом, а между одноименными уровнями разных узлов – протоколом.

Протокол определяет формат и очередность сообщений, которыми обмениваются два или более устройства, а также действия, выполняемые при передаче или приеме сообщений, а также при наступлении иных событий. Протоколы могут работать в рамках одной фирмы, консорциума разработчиков или всего сообщества разработчиков. Например, стандартная структура кадров в технологии Ethernet была предложена консорциумом фирм Digital Equipment Corporation (DEC), Intel и Xerox, а протоколы сети Интернет утверждаются Инженерным советом Интернета (Internet Engineering Task Force, IETF) – международным сообществом разработчиков.

Документы, создаваемые IETF, называютсяRFC (RequestsForComments — запрос для отзывов) иизначально предназначались для разрешения архитектурных проблем, возникавших в сетях-предшественницах Интернета. В настоящее время существует более 6000 различных документов RFC, которые стали фактическими стандартами Интернета, детально описывая все протоколы. Например, протокол HTTPописан в документе RFC2616, протоколIP– в документе RFC791и т.д.

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

Существуют три основные модели сетевоговзаимодействия:

1) модель, предложенная в 1982 г. международной организацией по стандартизации International Standards Organization(ISO)и называемая моделью взаимодействия открытых систем OSI (Open System Interconnection);

2) модель, предложенная в 1990 г. комитетом 802 Института инженеров по электротехнике и радиоэлектронике (IEEE), перед которым была поставлена задача стандартизации локальных и региональных сетей;

3) модель Интернета

### 1.8.1 Модель OSI

Модель OSIчасто называют эталонной моделью, так как она является результатом одной из первых попыток стандартизации стека сетевых протоколов.Модельохватывает все этапы, связанные с организацией обменов данными между удаленными устройствами, независимо от применяемого стека протоколов. Модель OSIне содержит описаний реализаций конкретного набора протоколов, ноопределяет различные уровни взаимодействия систем, даёт им стандартные имена и указывает, какие функции должен выполнять каждый уровень (рис.1.11).

Модель OSIсодержит семь уровней, каждый из которых реализует определенный этап, связанный с обменом данными. Нумерация уровней проводится снизу вверх. Три нижних уровня обычно реализуются аппаратным и программным обеспечением, остальныечетыре уровня реализуются программным обеспечением.

|  |
| --- |
| 7. Прикладной |
| 6. Представительный |
| 5. Сеансовый |
| 4.Транспортный |
| 3. Сетевой |
| 2. Канальный |
| 1. Физический |

Рис.1.11

Модель OSI описывает, каким образом информация проделывает путьчерез среду сети от одной прикладной программы к другой прикладной программе, выполняющейся на другом компьютере (рис.1.12).

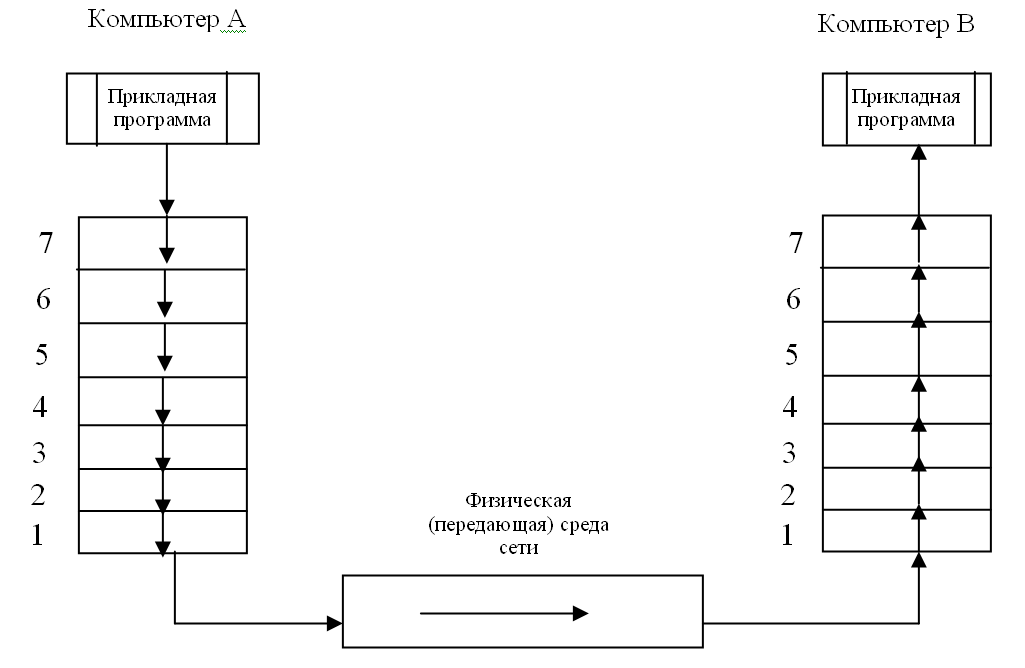


Рис.1.12

Данные, передаваемые каждым уровнем на нижележащий уровень сопровождаются служебной информацией текущего уровня. Нижележащий уровень воспринимает полученную информацию как самостоятельную единицу данных и при дальнейшей передаче добавляет к ней собственный заголовок.Процесс объединения заголовков и данных с последующей их передачей на нижележащий уровень называется **инкапсуляцией**. Некоторые уровни не требуют присоединения заголовков и просто выполняют преобразование полученных входных данных.

На приемном узле все действия выполняются в обратном порядке: на каждом уровне после получения входных данных проводится анализ служебной информации заголовка текущего уровня, после чего он отбрасывается и оставшаяся часть данных передается вышележащему уровню. Этот процесс называется **декапсуляцией**.

На рис.1.13 показан пример передачи строки «НГТУ» от одного компьютера другому.

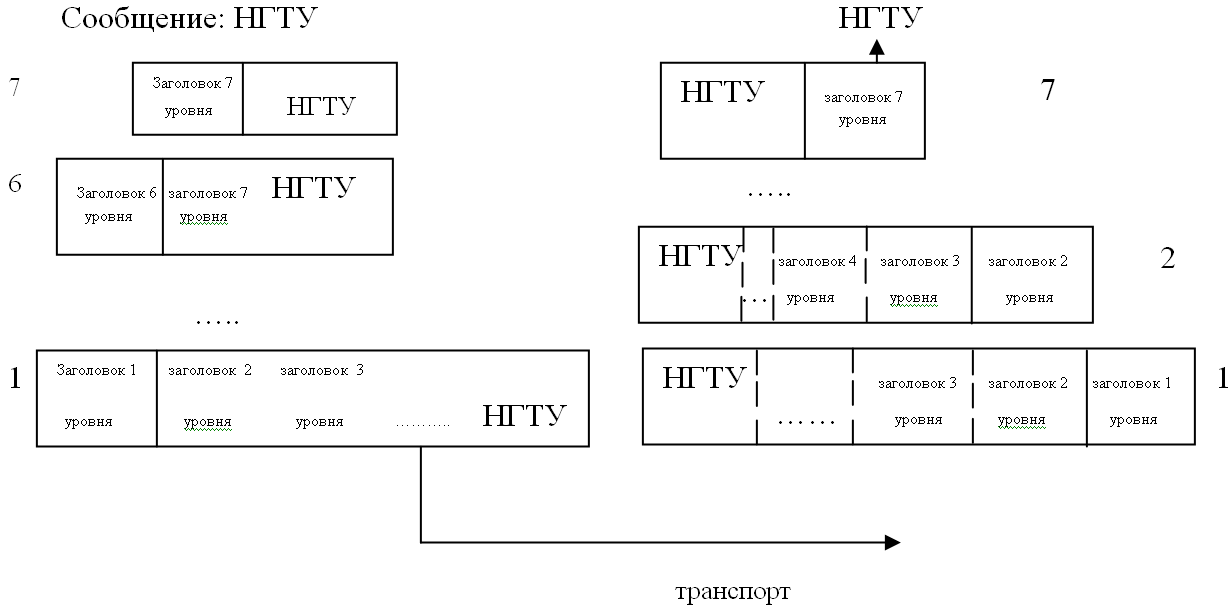
****

Рис.1.13

**Прикладной уровень**предназначен для поддержки сетевых приложений. Имеется множество протоколов прикладного уровня, с помощью которых сетевые приложения получают доступ к разделяемым ресурсам, таким как файлы, принтеры или гипертекстовые web-страницы, а также организуют совместную работу, например, с помощью протокола электронной почты. Единица данных, которой оперирует прикладной уровень, называется *сообщением*.Существует очень много различных служб прикладного уровня, например HTTP (передачаweb-страниц), FTP (передача файлов),SMTP (передача электронной почты).

**Представительный уровень** имеет дело с формой представления передаваемой по сети информации, не меняя при этом ее содержания. Этот уровень преобразует передаваемое сообщение в машинную форму, а на приемном узле выполняет обратное преобразование. При этом информация, передаваемая прикладным уровнем одной системы,всегда понятна прикладному уровню другой системы. Например, на передающем узле при передаче текстового сообщения может использоваться кодировка ASCII**,** а приемный узел может принимать это сообщение в кодировкеUTF-8.

На представительном уровне может выполняться шифрование и дешифрование данных, благодаря которому секретность обмена данными обеспечивается сразу для всех прикладных служб. Примером такого протокола является протокол SSL**,** который обеспечивает защищенный обмен сообщениями для протоколов прикладного уровня стека TCP/IP.

**Сеансовый уровень**отвечает за поддержку сеанса связи. Он управляет созданием и завершением сеанса,взаимодействием между сетевыми приложениями, синхронизацией задач, определением права на передачу данных. Например, во время видеоконференции обеспечивается синхронная передача двух потоков данных (аудио и видео) для каждого участника.

В настоящее время немногие приложения используют сеансовый уровень и он редко реализуется в виде отдельных протоколов, хотя функции этого уровня часто объединяют с функциями прикладного уровня и реализуют в одном протоколе.

**Транспортный уровень** обеспечивает передачупакетов данных между сетевыми приложениями, размещенными на удаленных узлах сети.Важно понимать, что сообщениями обмениваются не компьютеры, подключенные к сети, а приложения, работающие на каждом компьютере. Таких приложений одновременно может быть запущено множество и именно транспортный уровень отвечают за доставку сообщения нужному приложению.

В современных сетях используются два транспортныхпротокола –TCP и UDP. Протокол TCPсначала устанавливает логическое соединениес узлом-получателем и только после этого начинает передавать данные, обеспечивая высокую надежность за счет многоступенчатого контроля корректности принятых получателем данных. Протокол UDP обеспечивает передачу данныхбез предварительного установления логического соединения, при этом на приемной стороне могут возникать искажения и потерячасти данных.

Единица данных, которой оперирует транспортный уровень, для протокола TCPназывается *сегментом,* для протокола UDP *– дейтаграммой*.

**Сетевой уровень** обеспечивает передачу данных между узлами, находящимися в разных сетях, единицей данных этого уровня является *пакет*.

Сетевой уровень базируется на двух основных протоколах. Первый протокол называетсяIP (*InternetProtocol*) и определяет структуру передаваемых пакетови интерпретацию их содержимого маршрутизаторами и конечными узлами. Вторым протоколом является один из многочисленных протоколов маршрутизации, предназначенных для определения путипакетов от отправителя до адресатаи продвижения пакетов между сетями. Протокол IP и протоколы маршрутизации, несмотря на их функциональные различия,часто объединяют под общим именем IP, подчеркивая этим их связующую роль в организации глобальной сети.

Сетевой уровень использует сетевую адресацию и обеспечивает передачу пакета между отправителем и получателемчерез серию маршрутизаторов, используя службы канального уровня.

**Канальный уровень**организует адресацию узлов внутри одной локальной сети и обмен данными между этими узлами. Протоколы канального уровня работают только в сетях с базовойтопологией (звезда, кольцо, шина) иориентированы на определенный типфизической среды передачи данных. Одной из основных функций канального уровня является получение доступа к среде передачи данных. Единица данных этого уровня называется *кадр*

Примерами протоколов канального уровня являются, например,Ethernet,IEEE 802.11, РРР.

**Физический уровень**обеспечивает передачу между узлами отдельных битов информации. Протоколы физического уровня напрямую зависят от физической среды передачи данных, так как механизмы передачи бита для витой пары, оптоволокна или радиоканала различны.

На физическом уровне определяются тип кодирования,характеристикиэлектрических сигналов (требования к фронтам импульсов, уровням напряжения или тока передаваемого сигнала) и стандартизируются типы разъёмов и назначение их каждого контакта (например, RJ-45).

В зависимости от типа коммуникационное устройство может работать либо только на физическом уровне (повторитель), либо на физическом и канальном (мост и коммутатор), либо на физическом, канальном и сетевом (маршрутизатор).

Функции всех уровней модели OSI могут быть отнесены к одной из двух групп:

* зависящие от конкретной технической реализации сети;
* ориентированные на работу с приложениями.

Три нижних уровня (физический, канальный и сетевой) являются cетезависи-мыми, так как протоколы этих уровней тесно связаны с технической реализацией сети и используемым коммуникационным оборудованием. Например, переход на оборудование FDDI (волоконно-оптический интерфейс передачи данных) означает полную смену протоколов физического и канального уровней во всех узлах сети.

Три верхних уровня (прикладной, представительный и сеансовый**)** ориентированы на приложения и мало зависят от технических особенностей построения сети. На протоколы этих уровней не влияют изменения в топологии сети, замена оборудования или переход на другую сетевую технологию. Поэтому переход от Ethernet 10BASE-Tна GigabitEthernet 1000BASE-Tне потребует изменений в программных средствах, реализующих функции прикладного, представительного и сеансового уровней.

Транспортный уровень является промежуточным, он скрывает все детали функционирования нижних уровней от верхних. Это позволяет разрабатывать приложения, не зависящие от технических средств непосредственной транспортировки сообщений.

В таблице 1.3 приведенырабочие области основных коммуникационных устройств и их соответствие уровням модели OSI.

Таблица 1.3

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Уровень модели OSI | Устройство | Реализация | Рабочая область | Единица данных |
| физический | повторитель | аппаратная | физический сегмент сети | поток битов |
| канальный | мост, коммутатор | аппаратная | физические сегменты сети | кадр |
| сетевой | маршрутизатор | аппаратная или программная | сети, подсети | пакет |
| транспортный | – | программная | сети, подсети | сегмент, дейтаграмма |
| прикладной | шлюз | аппаратная или программная | интерсети | сообщение |

### 1.8.2Модель TCP/IP

Модель TCP/IP используется глобальной сетью Интернет и является частным случаем эталонной модели OSI. Это связано с двумя причинами:

* сеть Интернет использует в основном модель взаимодействия «клиент-сервер», основанную на том, что приложение-клиент обращается с определенным запросом к приложению-серверу. Сервер обрабатывает запрос и возвращает клиенту результат обработки. В этом случае установление связи и последовательность обмена данными жестко определена и отпадает необходимость использования сеансового уровня.
* форма представления данных в сети Интернет стандартизована протоколом HTTP, поэтому становится ненужным представительный уровень.

Поэтому модель TCP/IPсодержитвсего пять уровней – прикладной, транспортный, сетевой, канальный и физический.

### 1.8.3 Модель IEEE 802

МодельIEEE 802 ориентирована на локальные сети и описывает только средства канального уровня, разделяя его на два независимых подуровня – логический и управления доступом к среде передачи данных. Логический подуровень отвечает за передачу кадров между узлами сети с различной степенью надёжности, а также реализует функции интерфейса с прилегающим к нему сетевым уровнем.

Подуровень управления доступомуправляет доступом к среде передачи данныхи появился из-за существования в локальных сетях разделяемой среды передачи данных. Этот подуровень обеспечивает корректное совместное использование общей среды, предоставляя её в соответствии с определённым алгоритмом в распоряжение узлов сети.

## 1.9 Практическаяработа

### 1.9.1Общие сведения

На рис. 1.14. приведена структура локальной сети факультета, которая состоит из двух логических сегментов, соединенных между собой через маршрутизаторы.Функции маршрутизаторов выполняют коммутаторы третьего уровня Cisco 3560G, каждый из которых имеет по 48 портов.Сегменты построен на коммутаторах Cisco 1800G, имеющих по 24 порта.

К коммутаторам подключаются рабочие станции (РС) пользователей, расположенные в терминальных классах (ТК), на кафедрах, в деканате и т.д. Для каждого помещения указано название и количество РС, например, на рис. 1.15 показано обозначение терминального класса ТК 203а, в котором находится 13 РС.



Рис. 1.15

В одном из сегментов имеется мощная вычислительная блейд-система HP c7000, состоящая из 16 блейд-серверов, а также 6 серверов HP Proliant DL350 G5, на которых реализован кластер виртуальных машин, используемых в учебном процессе (например, веб-серверы students.ami.nstu.ru, terminal.ami.nstu.ru и др.).

#### 

#### Рис.1.14

### 1.9.2Задание

1. Выполните анализ локальной сети факультета по следующим пунктам, все результаты занести в отчет:
   * укажитекоммуникационные устройства, используемые в сети, и найдитев Интернете их основные технические параметры;
   * какие линии связи используются в сети;
   * схема соединения Вашей рабочей станции(РС*i)* с сервером fpm2.ami.nstu.ru;
   * укажите структуру сетевого программного обеспечения согласно модели OSI на каждом узле схемы соединения РС*i* с сервером fpm2.ami.nstu.ru;
2. Определите IP и MAC-адреса Вашей рабочей станции и сервера fpm2.ami.nstu.ru. Обратите внимание на то, что рабочие станции работают под управлением ОС Windows, а сервер – под управлением ОС Linux. Также напомним, что у одного компьютера может быть несколько сетевых интерфейсов, каждый из которых имеет собственные параметры.
3. Выполните трассировку маршрута передачи пакетов от РС*i*к серверу fpm2.ami.nstu.ru и в обратном направлении, найдите количество промежуточных узлов и их IP-адреса. Поясните причину различия IP-адресов промежуточных узлов при прямой и обратной трассировках. Для лучшего понимания механизма трассировки эту часть задания желательно выполнять в компьютерном классе.

## 1.10Контрольные вопросы

1. Модели сетевого взаимодействия и их основные свойства.
2. Понятия сетевого протокола и интерфейса.
3. Стек протоколов сети Интернет: понятие, основные протоколы стека.
4. В чем преимущества и недостатки многоуровневой организации системы?
5. Уровни стека протоколов Интернет и их основные функции.
6. Протокольные единицыданных разных уровней в стеке протоколов.
7. Виды сетевого оборудования, их назначение.
8. Укажите уровни модели OSI, на которых работают различные коммуникационные устройства.
9. Физические среды передачи данных и их особенности.
10. Инкапсуляция: понятие и механизм реализации.
11. Какая топология используется в локальной сети факультета.
12. Как реализован выход локальной сети в корпоративную сеть университета ?

2. ТЕХНОЛОГИИ КАНАЛЬНОГО УРОВНЯ

*Прочитав эту главу, вы узнаете:*

* *стандарты и оборудование канального уровня;*
* *основы технологии Ethernet;*
* *типы кадров Ethernet и их структуры;*
* *определение следующих терминов: разделяемая среда, контрольная сумма,мост, коммутатор, коллизия,таблица коммутации*

## 2.1Стандарты канального уровня

Основными функциями канального уровня на стороне отправителя являются формирование кадров из пакетов, полученных от сетевого уровня, и передача их на физический уровень. Кадры формируются из полученных пакетов путем добавления к ним служебной информации – ограничительных меток начала и конца кадра, адресов получателя и отправителя, а также данных, необходимых для обнаружения возможных ошибок в переданном пакете на приемной стороне.Формат кадра,под которым понимается состав и порядок следования отдельных полей кадра, определяется используемымпротоколом.

На стороне получателя канальный уровень формирует кадры из полученной с физического уровня последовательности битов путем выделения из нее ограничительные меток и других полей в соответствии с протоколом. Далее из сформированного кадра выделяется пакет данных, который передается на сетевой уровень для дальнейшей декапсуляции, все служебные поля кадра при этом отбрасываются.

Стандарты семейства IEEE 802.x охватывают только два нижних уровня модели OSI – канальный и физический, так как именно эти уровни в наибольшей степени отражают специфику локальных сетей. Уровни, расположенные выше, имеют общие черты, характерные как для локальных, так и глобальных сетей.

СтандартыIEEE 802.x разделяют канальный уровень на два подуровня–логический и управление доступом к среде передачи данных (рис.2.1).Логический подуровеньLLC (LogicalLinkControl) выполняет формирование кадров, реализует функции инкапсуляции и декапсуляции, а также взаимодействует с сетевым уровнем.Здесь же может обеспечиваться различный уровень контроля ошибок в передаваемых данных.

Логические процедуры передачи кадров и связь с сетевым уровнем(IEEE 802.2)

Семейство технологий Ethernet

(IEEE802.3)

TokenRing

(IEEE802.5)

Семейство технологий WiFi

(IEEE802.11)

Сетевой уровень

Физический уровень

Подуровень LLC

Подуровень MAC

Рис. 2.1

В общем случаепрограммное обеспечение канального уровня должно обеспечивать:

* добавление к передаваемым данным уникальных ограничительных меток с высокой надежностью их распознавания;
* поддержку системы адресации согласно используемому протоколу;
* распознавание ошибок и, по возможности, их исправление.

### 2.1.1 Подуровень LLC

Протокол LLC предусматривает несколько режимов работы, согласно которым кадры передаются либо дейтаграммным способом, либо с помощью процедур с установлением соединения между взаимодействующими узлами сети и,при наличии искажений,восстановлением кадров путем их повторной передачи.

Различают три режима работы протокола LLC:

• LLC1 – процедура без установления соединения и без подтверждения. Этот дейтаграммный режим работы используется, когда контроль и восстановление данных после ошибок и упорядочение данных проводится протоколами вышележащих уровней;

• LLC2 – процедура с установлением соединения и подтверждением. При этом перед началом передачи между отправителем и получателем устанавливается логическое соединение, а затем в ходе обмена данными выполняются процедуры восстановления кадров после ошибок и упорядочения потока кадров в рамках установленного соединения;

• LLC3 – процедура без установления соединения, но с подтверждением. Применяется, когда временные задержки, связанные с установлением соединения, не допускаются, но подтверждение о корректности приема данных необходимо. Протокол LLC3 используется в сетях, работающих в режиме реального времени.

Во время передачи данных всегда есть вероятность возникновения ошибок, связанных с потерей или искажением битов. Для обнаружения ошибок могут применяться различные способы, например, использование бита чётности или контрольное суммирование.Бит четности – это дополнительный бит, добавляемый к битам передаваемых данных так, чтобы число единиц в передаваемом байте всегда было четным или нечетным.

Контрольная сумма – это целое число, значение которого зависит от значения каждого бита данных, т.е. при изменении даже одного бита значение контрольной суммы изменится. Контрольная сумма вычисляется на стороне отправителя и включается в состав кадра, а на приемной стороне проводится повторное вычисление контрольная сумма. При отсутствии ошибок эти значения должны совпадать.

Указанные режимыпротокола LLC являются общими для всех методов доступа к передающей среде, определенных стандартами IEEE 802.х.

### 2.1.2 Подуровень MAC

Подуровень MAC появился из-за существования в локальных сетях разделяемой среды передачи данных. Этот подуровень обеспечивает корректное совместное использование общей среды, предоставляя её в распоряжение того или иногоузла сети в соответствии с определённым методом доступа.

Методы доступак передающей среде - это совокупность правил, по которым узлы сети получают доступ к физической среде передачи данных. Они определяют, какой из узлов может следующим использовать эту среду. Методы доступа реализуются протоколами передачи данных нижнего уровня и являются одним из основных признаков, по которым различают сетевое оборудование.

Возможны следующие методы доступа:

* методы, основанные на резервировании времени, когда любойузел сети осуществляет передачу только в течение заранее зарезервированных для него временных интервалов (слотов);
* селективные методы, когда узел осуществляет передачу только после получения разрешения, направляемого каждомуузлу по очереди центральным управляющим органом сети (алгоритм циклического опроса), или передаваемого от станции к станции (алгоритм передачи маркера);
* методы случайного доступа, когда каждыйузел пытается «захватить» передающую среду.

В локальных сетях получили распространение несколько MAC-протоколов**,** реализующих различные алгоритмы доступа к разделяемой среде. Эти протоколы полностью определяют специфику таких технологий, как Ethernet, FastEthernet, GigabitEthernet, WiFi, TokenRing и т.д. Каждый протокол уровня MAC может использоваться с любым протоколом уровня LLC и наоборот.

Общепринятого соглашения в отношении формата кадров не существует. В каждой конкретной сетевой технологии задаётся собственный формат, но внутреннее содержание кадра при этом не меняется. В настоящее время в локальных сетях наиболее часто применяется технология Ethernet, поэтому далее будем рассматривать особенности именно этой технологии.

### 2.1.3 Адресация канального уровня

Для идентификации сетевых интерфейсов используются регламентированные стандартом IEEE 802.3 уникальные 6-байтовые MAC-адреса, которые могут быть индивидуальными, групповыми или широковещательными.

Первый (младший) бит старшего байта адреса указывает, является этот адрес индивидуальным или групповым. Если первый бит MAC- адреса равен нулю, то это индивидуальный адрес, если равен единице –групповой. Групповая рассылка производится конкретной группе узлов сети, формируемой администратором сети. Если какой-либо сетевой интерфейс включен в группу, то он имеет два MAC-адреса – индивидуальный и групповой.

Групповой адресFF:FF:FF:FF:FF:FF, все биты которого равны единице, идентифицирует все узлы сети и называется широковещательным.

Таким образом, если в кадре в качестве адреса получателя указан индивидуальный адрес, то этот кадр будет получен только одним узлом сети. Если указан групповой адрес, то кадр получат все узлы, входящие в заданную группу, а при указании широковещательного адреса кадр будет получен всеми узлами локальной сети.

Второй бит старшего байта указывает способ назначения адреса. Нулевое значение соответствует централизованному адресу, который назначаетсяпроизводителем оборудования и содержит код производителя (3 старших байта) и уникальный номер сетевого интерфейса(3 младших байта). Таким образом каждый производитель может выпустить около 16 миллионов сетевых интерфейсов под одним идентификатором организации. Единичное значение второго бита говорит о том, что MAC-адрес является локальным и назначается администратором сети.

В стандартах IEEEEthernet младший бит байта отображается в крайне левой позиции байта, а старший бит – в правой. Такой способ отображения называется бит-реверсным и соответствует порядку передачи битов в линию передатчиком Ethernet. Содержимое байтов при записи MAC-адреса в нотации IEEE разделяется символом двоеточия «:».

В стандартах других организаций содержимое байтов отображается в каноническом виде (старший бит слева), а в качестве разделителя байтов используется символ «-».

## 2.2Основы технологии Ethernet

Первая версия протокола Ethernet была разработана фирмой Xerox в 1975 г. в виде внутреннего стандарта, предусматривающего передачу данных со скоростью до 3 Мбит/с. В 1980 г. фирмы DEC, Intel и Xerox совместно разработали и опубликовали стандарт Ethernet 2 для сети, построенной на основе коаксиального кабеля cо скоростью передачи до 10 Мбит/с. Эту версию стандарта назвали аббревиатурой по наименованию фирм-разработчиков Ethernet DIX или Ethernet 2.

На основе стандарта Ethernet DIX комитетом IEEE802 было разработаноцелое семейство международныхстандартов, часть из которых представлена в таблице 2.1.В настоящее времясемейство Ethernet насчитывает свыше 30 стандартов, отличающихся скоростью и типом среды передачи.

Таблица 2.1

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Номер IEEE | Название | Год | Скорость передачи | Физическая среда | Базовая топология | Примечание |
| 802.3 | 10BASE-5 | 1983 | 10 Мбит/с | коаксиальный кабель | общая шина | Добавлен LLC-заголовок IEEE 802.2, следующий за заголовком IEEE 802.3 |
| 802.3i | 10BASE-T | 1990 | 10 Мбит/с | витая пара | звезда |  |
| 802.3y | 100BASE-T2 | 1998 | 100 Мбит/с | витая пара 3 кат. | звезда |  |
| 802.3z | 1000BASE-T2 | 1998 | 1 Гбит/с | оптоволокно | звезда |  |
| 802.3ab | 1000BASE-T | 1999 | 1 Гбит/с | витая пара 5 кат. | звезда |  |
| 802.3an | 10GBASE-T | 2006 | 10 Гбит/с | витая пара 6 или 7 кат. | звезда |  |
| 802.3ba | 100 Gigabit Ethernet | 2010 | 100 Гбит/с | оптоволокно | звезда |  |
| 802.3bs | 400 Gigabit Ethernet | 2017 | 400 Гбит/с | оптоволокно | звезда | Имеет 4 реализации |

Ранние версии Ethernet (до1990 г.), рассчитанные на работу с коаксиальным кабелем, могли работать только в сетях с топологией «общая шина». При этомобщая шина является разделяемым ресурсом и кадр, передаваемый любым узлом сети, принимается всеми другими узлами сети. Более поздние стандарты ориентированы на отказ от общей шины и использование в качестве среды передачи витой пары или оптоволоконного кабеля.

Протоколы Ethernet, работающие на скорости до 10 Гбит/с, используют метод случайного доступа к разделяемой среде, называемый множественным доступом с прослушиванием несущей и обнаружением коллизий (CSMA/CD, Carrier Sense Multiple Access with Collision Detection).

Алгоритм доступа к разделяемой среде у протоколов Ethernet следующий:

* узел, который должен начать передачу кадра проверяет занятость разделяемой среды, проверяя наличие несущей частоты передаваемых сигналов в общей шине;
* если несущая частота отсутствует, то узел начинает передачу, иначе он переводится в состояние ожидания;
* в состоянии ожидания узел делает случайную паузу длительностью Tп = L \* To, где To – интервал отсрочки, равный 512 битовых интервалов, L – целое число, выбранное случайным образом из диапазона [0, 2n ], где n – номер повторной попытки передачи данного кадра и может принимать значения от 1 до 10. После 10-й попытки пауза не увеличивается, а после 16 попыток кадр отбрасывается.
* после передачи любого кадра все узлы выдерживают паузу в 96 бит-тактов (9,6 мксек для скорости 10 Мбит/с) для приведения сетевых адаптеров в исходное состояние.

Недостатком метода CSMA/CD является возможность того, что два узла сети могут начать передачу своих кадров одновременно, что приводит к искажению последовательности битов, передаваемых этими узлами. Такая ситуация называется коллизией, результатом которой является «пробка» – короткая последовательность битов с хаотическим распределением единиц и нулей. «Пробка» распространяется по всей сети, ее получают все узлы, включая и те, которые только что отправили свои кадры. Для них это сигнал о том, что отправленные кадры потеряны, и необходимоповторить их передачу.

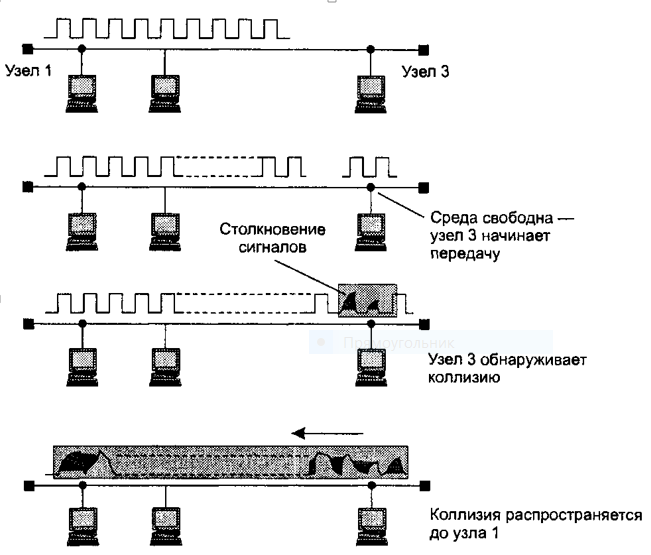


Рис.2.2

На рис. 2.2 показан пример возникновения коллизии. Здесь узел 1 после проверки занятости среды начинает передачу своего кадра и электрические сигналы начинают распространяться по общей шине слева направо. В этот же момент времени узел 3 проводит проверку занятости шины. Он видит, что шина свободна, так как сигнал от узла 1 еще не успел до него дойти, и начинает передачу своего кадра, который распространяется справа налево и в некоторый момент времени эти сигналы перемешиваются, т.е. возникает «пробка», которая перемещается по шине справа налево до узла 1. После определения коллизии каждый из передающих узлов должен прекратить передачу своих кадров и перейти в состояние ожидания.

Параметры протокола Ethernet подобраны таким образом, чтобы коллизия обнаруживаласьвсеми узлами, включая самые удаленные друг от друга.

### 2.2.1 Форматы кадровEthernet

В сетях Ethernet на канальном уровне используются четыре различных форматакадров. Это связано с историей развития технологии Ethernet, имеющей период существования до принятия стандартов IEEE802, когда подуровень LLC не выделялся из общего протокола и, соответственно, заголовок LLC не применялся. Один и тот же тип кадра может иметь разные названия, поэтому для каждого типа кадра ниже приведено по нескольку наиболее употребительных названий:

* кадр Ethernet DIX (Ethernet 2);
* кадр 802.3/LLC (802.3/802.2);
* кадр Raw 802.3 (Novell 802.3);
* кадр Ethernet SNAP.

На рис.2.3 приведена классификация форматов кадров Ethernet.Первым типом кадра исторически является **Ethernet DIX**, формат которого был разработан еще до появления стандарта IEEE802.3. Основнымиего отличиями от других типов кадров являются:

* отсутствие разделения канального уровня на подуровни, поэтому его поля поддерживают функции LLC и MAC(поле T относится к LLC, остальные - к MAC);
* явное указание протокола, который вложил в кадрсвои данные.

Тип кадра

С явным указанием типа

С неявным указанием типа

DIX (Ethernet 2)

LLC

RAW

SNAP

Рис. 2.3

На рис.2.4 приведен формат кадра EthernetDIX. Здесь полепреамбулысодержит семь синхронизирующих байтов 10101010 и начальный ограничитель кадра*,* состоящий из одного байта 10101011. После ограничителя следует первый байт заголовка кадра. Конечный ограничитель кадра современными протоколами не используется.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Заголовок кадра** | | | **Тело кадра** |  |
| 8 | 6 | 6 | 2 |  | 4 |
| Преамбула | Адрес  получателя | Адрес  отправителя | Код протокола | Поле данных(46 – 1500 байтов) | Контроль- ная сумма |

Рис.2.4

Поля «Адрес получателя» и «Адрес отправителя» содержат MAC-адреса сетевых интерфейсов получателя и отправителя соответственно.

Поле «Код протокола» содержит двухбайтовый Ethernet-код протокола верхнего уровня, данные которого вложены в кадр (например, для IPv4 – 0x0800, для ARP – 0x0806, для IPv6 –0х08DD, для IPX – 0х8137 и т.д.).

Поле данных содержит данные объемом от 46 до 1500 байтов, принимаемые отпротокола верхнего уровня; если длина данных меньше 46 байт, то поле дополняется до минимального размера нулевыми байтами заполнения для надежного распознавания коллизий.

Поле «Контрольная сумма» содержит значение контрольной суммы, вычисляемое по алгоритму CRC-32.

**Кадр 802.3/LLC**появился в результате разделения канального уровня на подуровни LLC иMAC, в связи с чем кадр подуровня LLCдолжен быть вложен в кадр подуровня MAC. В результате инкапсуляции формат кадра 802.3/LLCпринял вид, показанный на рис.2.5.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **MAC-заголовок** | | | **LLC-заголовок** | | | **Тело кадра** |  |
| 8 | 6 | 6 | 2 | 1 | 1 | 1 (2) |  | 4 |
| Преам-була | Адрес  получателя | Адрес  отправителя | Длина | DSAP | SSAP | Control | Поле данных, размер 46 – 1497(1496) байтов | Контроль- ная сумма |

Рис.2.5

Здесь поле «Длина» содержит суммарный размер LLC-заголовка и поля данных.Так как кадр LLC имеет заголовок длиной 3 (в режиме LLC1) или 4 байт (в режиме LLC2), то максимальный размер поля данных уменьшается до 1497 или 1496 байт.

ПоляSSAP и DSAP предназначены для хранения однобайтовых IEEE-кодов протоколов источника (SourceService Access Point) и назначения (DestinationService Access Point) для подуровня LLC. Таким образом разработчики стандарта IEEE802.3 заложили возможность передачи входного пакета между двумя различными протоколами.В подавляющем большинстве случаев значения SSAP и DSAP совпадают.

Поле Control используется для обозначения типа кадра данных — информационный, управляющий или ненумерованный. В этом поле указываются порядковые номера отправленных и успешно принятых кадров, если подуровень LLC работает по процедуре LLC2 с установлением соединения.

*Обратите внимание: IEEE-коды протоколов отличаются от кодов протоколов, используемых в кадрах EthernetDIX. Например IEEE-код протокола IP равен 0х06, для SNAP – 0xAA, для STP – 0x42.*

**Кадр Raw 802.3**использовалсяв стеке протоколов IPX/SPXфирмы NovellInc. и представлял собой кадр 802.3/LLCбез LLC-заголовка. Отсутствие потребности в этом заголовке объясняется тем, что в качестве протокола сетевого уровня в стеке IPX/SPX мог применяться единственный протокол IPX. В результате нескольких поглощений фирма Novellпрекратила существованиев 2014 г.

**Кадр EthernetSNAP**был предложен комитетом IEEE802 с цельюиспользования протокола доступа к подсетям SNAP (Subnetwork Access Protocol), который позволяет увеличить число протоколов, инкапсулируемых вподуровень MAC. Заголовок SNAPимеет размер пять байтов и содержит трехбайтовый уникальный код организации-разработчика протокола (Organizationally Unique Identifier, OUI) и Ethernet-код протокола (Protocol Identification, PI). Например, для протоколов, контролируемых IEEE,значение OUIравно 0х000000, а для протоколов Cisco – 0x00000С.

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | **MAC-заголовок** | | | **LLC-заголовок** | | | **SNAP-заголовок** | | **Тело кадра** |  |
| 8 | 6 | 6 | 2 | 1 | 1 | 1 (2) | 3 | 2 |  | 4 |
| Пре-амбула | Адрес  получателя | Адрес  отправителя | Дли-на | DSAP | SSAP | Control | OUI | PI | Данные(46 – 1492) | Контроль ная сумма |

Рис.2.6

С помощью заголовка SNAP достигнута совместимость с кодами протоколов, используемых в кадрах Ethernet DIX, а также создана универсальная схема кодирования протоколов.

Современные сетевые интерфейсы могут работать со всеми типами кадров технологии Ethernet, используя автоматическое распознавание типа. Однако в настоящее время основным типом кадров по прежнему является EthernetDIX.

Распознавание начинается с анализа 13 и 14 байта кадра. Если значение, записанное в этих байтах, превышает значение максимальногоразмера поля данных,равное 1500 (0x05DC), значит в этом поле записан Ethernet-код вышестоящего протокола и этот кадр является кадром EthernetDIX.

Дальнейшее распознавание типа кадра проводится по наличию или отсутствию полей LLC. Поля LLC могут отсутствовать только в том случае, если за полем длины идет начало пакета IPX, а именно 2-байтовое поле контрольной суммы пакета, которое всегда равно 0хFFFF. Ситуация, когда поля DSAP и SSAP одновременно содержат такое значение, возникнуть не может, поэтому наличие двух байтов (15 и 16) со значением 0хFFговорит о том, что это кадр Raw 802.3.

В остальных случаях дальнейший анализ проводится в зависимости от значений полей DSAP и SSAP. Если они равны 0xАА, то это кадр Ethernet SNAP, а если нет, то 802.3/LLC.

## 2.3 Оборудование канального уровня

В локальных сетях на канальном уровне работают мосты (bridge) и коммутаторы (switch).Мост имеетнесколько портов и обеспечиваетпродвижение кадров и изоляцию трафика между сегментамив локальной сети. Напомним, что сегмент – это логически обособленная часть сети, не содержащая мостов и коммутаторов. Узлы, входящие в сегмент, и соответствующий порт моста взаимодействуют по схеме «многие к одному».

Мост содержит микропроцессор и встроенную память, в которой хранится адресная таблица, содержащая соответствие MAC-адреса узла номеру порта, к которому подключен этот узел. При получении на любой порт кадра мост распознает MAC-адрес получателя и пытается найти его в адресной таблице. Если адрес получателя не найден или является широковещательным, то кадр направляется на все порты, кроме порта отправителя, а адрес отправителя и номер подключенного к нему порта записываются в адресную таблицу.

Если адрес найден, отправитель и получатель находятся в одном сегменте, то кадр удаляется из буфера (фильтруется), а если адрес найден и кадр не принадлежит сегменту отправителя, то он передается в требуемый сегмент.

Если порт пункта назначения в данный момент занят, то мост временно сохраняет кадр в памяти до освобождения порта. Мост также вычисляет контрольную сумму (CRC) перед отправкой кадра получателю.

Таким образом мост передаёт данные только непосредственно получателю, исключениями являются широковещательный трафик и трафик для узловс неизвестнымMAC-адресом получателя. Это повышает производительность и безопасность сети, избавляя остальные сегменты сети от необходимости обрабатывать данные, которые им не предназначались.

Порты моста не имеет собственного MAC-адреса, т.к. являются узлами соответствующих сегментов и обрабатывают все приходящие кадры независимо от их адреса назначения.

На рис. 2.7 приведен пример адресной таблицы моста, подключенного к двум сегментам сети (LANA и LANB), сегмент LANA содержит 3 узла, а LANB содержит 2 узла. Адреса, помеченные звездочками, являются статическими (назначены вручную), остальные адреса динамические. У адреса, помеченного символом “+” истек срок жизни.

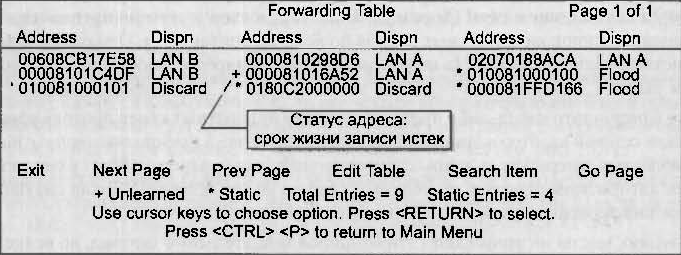


Рис. 2.7

Поле Dispn (disposition) по умолчанию содержит порт назначения (в нашем случае – имя сегмента), но в ручном режиме здесь можно указать способ фильтрации кадров. Например, Discardпредписывает не передавать кадр с указанным адресом в сегмент назначения, а Flood – передавать кадр в широковещательном режиме даже если адрес получателя не является широковещательным.

Процессор моста последовательно обрабатывает все кадры, поступающие одновременно на разные порты. Это является существенным недостатком и является причиной низкой пропускной способности. В настоящее время мосты не используются и повсеместно заменяются коммутаторами.

**Коммутатор**по сути является многопроцессорным мостом. Здесь для каждого порта имеется собственныймикропроцессор, реализующий все алгоритмы работы моста. Это дает возможность одновременно обрабатывать кадры, поступающие на все порты. При этом по каждому порту кадры будут передаваться с максимальной скоростью, на которую рассчитан протокол.

Коммутатор хранит в памяти таблицу коммутации, в которой указывается соответствие MAC-адреса узла порту коммутатора. При включении коммутатора эта таблица пустая и он некоторое время работает в режиме обучения, заполняя таблицудля всех активных MAC-адресов, после чего в результате трафик локализуется.

Основными параметрами коммутаторов являются:

* количество портов;
* скорость продвижения (кадров в секунду) - скорость, с которой выполняются операции приема кадра в буфер, просмотр адресной таблицы и передача кадра в сеть через найденный в таблице порт;
* пропускная способность (Мбит/с) – количество **пользовательских** данных, переданных в единицу времени через порты, максимальное значение производительности достигается на кадрах максимальной длины;
* размер таблицы коммутации – определяет число MAC-адресов, с которыми одновременно может работать коммутатор.

Коммутаторы могут продвигать миллионы кадров в секунду в то время, когда скорость продвижения для мостов составляет до 5000 кадров/сек. Общую пропускную способность коммутатора можно оценить следующим образом: P = (N/2) \* M, где P – пропускная способность коммутатора, N – количество портов, M – максимальная скорость протокола (Мбит/с). Максимальный размер таблицы зависит от назначения коммутатора и может составлятьот 8192 адресов (коммутаторы для рабочих групп и малых офисов) до 65536 адресов (коммутаторы магистралей сетей).

Кроме основной функции(фильтрации трафика с максимальной скоростью, предусмотренной используемым протоколом)современные коммутаторы имеют ряд дополнительных функций:

* **маршрутизация** - управление коммутацией пакетов на сетевом уровне моде-ли OSI, т.е. между сетями. Обычно такие коммутаторы называют коммутаторами 3-го уровня.
* **создание виртуальных сетей**, под которыми понимаются группы узлов сети, трафик которых, в том числе широковещательный, на канальном уровне пол-ностью изолирован от трафика других узлов сети;
* **приоритезация трафика –** назначение приоритетов различным видам трафика для более эффективного обслуживания пользователей сети. Например, допустимые требования по задержкам кадров аудио- и видеотрафика более жесткие, чем для передачи файлов.
* **стекирование** - соединение двух или более управляемых коммутаторов, предназначенное для увеличения числа портов; при этом полученная группа идентифицируется остальными сетевыми устройствами как один логическийкоммутатор, имеющий один IP-адрес (для коммутаторов 3 уровня) и один MAC-адрес
* **поддержка древовидной топологии сети** в случае наличия в ней резервных связей.

Появление последней функции связано с тем, что алгоритм работы мостаможет надежно работать только в сетях, имеющих древовидную топологию, в которой отсутствуют связи между отдельными сегментами.При этом отказ любой линии связи или коммутатора приводит к потере связности сети, которая фактически распадается на несколько сегментов.Для увеличения надежности коммутируемой сети администратор может формировать между узлами сети резервные связи, которые в штатном режиме необходимо вручную отключать путем блокирования соответствующих портов.

Наличие избыточных резервных связи приводит к образованию петель при продвижении кадров, что приводит к бесконечной циркуляции кадра по петле, появлению нескольких копий одного кадра и постоянной перестройке мостами и коммутаторами своих адресных таблиц.Решить эту проблему позволяет функция автоматическогоконфигурированиядревовидной топологии сети в случае отказа линий связи или коммутатора.

В терминальных устройствах сети функции канального уровня реализуются **сетевыми адаптерами и их драйверами**.

В современных локальных сетях в реализации канального уровня имеются следующие отличия от ранних сетей:

* отказ от применения кабельной разделяемой среды передачи данных и переход к коммутируемым сетям, в которых все узлы подключаются к коммутаторам индивидуальными линиями связи;в этом случае каждый сегмент сети состоит только из одного узла, а сам узел и соответствующий порт коммутатора взаимодействуют по схеме точка-точка;
* применение дуплексного (двухстороннего) режима передачи данных.

*Обратите внимание: все протоколы канального уровня обеспечивают доставку данных между двумя любыми узлами только в сетях с базовой топологией, к которым относятся общая шина, кольцо и звезда.*

## 2.4Практическая работа

1. Разработать и отладить программу, выполняющую анализ потока кадров. Потокикадров представлены в виде двоичныхфайлов, местонахождение которых уточняется у преподавателя. В кадрах отсутствует преамбула и контрольная сумма, для исходящего кадра длина может быть меньше минимальной.

Требования к программе:

* предусмотреть возможность ввода имени файла с клавиатуры;
* обеспечить вывод на экран для каждого кадра следующей информации: номер кадра, размер, тип, MAC- адреса отправителя и получателя, IP- адреса отправителя и получателя (находятся в заголовкеIP-пакета).
* обеспечить вывод итоговых результатов обработки файла – общее количество обработанных кадров,количество кадров каждого типа (EthernetDIX, 802.3/LLC, Raw 802.3,EthernetSNAPи т.д.), количество кадров с вложением протоколов IPиARP, структуры которых можно найти в разделе 3.

2. Выполнить полный анализ кадра с номером Х, где Хпоследняя цифра в номере Вашей зачетной книжки. Анализ можно провести с помощью любого 16-ричного редактора, например, свободно распространяемых редакторовHxD или DMDE.

## 2.5Контрольные вопросы

* 1. Поясните причину разбиения канального уровня на подуровни.
  2. Подуровень LLC: назначение, режимы работы.
  3. Подуровень MAC: назначение, методы доступа к передающей среде.
  4. Структура MAC-адреса.
  5. Алгоритм доступа к разделяемой среде в протоколах Ethernet.
  6. Коллизии Ethernet: причины возникновения, методы борьбы с ними.
  7. Классификация форматов кадров Ethernet.
  8. Алгоритм определения типа кадров Ethernet
  9. Мост: назначение, алгоритм работы.
  10. Коммутатор: назначение, основные параметры, дополнительный функции.

3. ТЕХНОЛОГИИ СЕТЕВОГО УРОВНЯ

*Прочитав эту главу, вы узнаете:*

* *системуадресации сетевого уровня;*
* *структуру IP-пакета;*
* *основы IP-маршрутизации*
* *определение следующих терминов: маршрутизация, продвижение данных, маска сети, время жизни, метрика, таблица маршрутизации.*

Основными функциями сетевого уровня являются продвижение пакетов между сетями и маршрутизация. Здесь под сетью понимается любая локальная сеть, то есть совокупность ТУ, соединённых между собой согласно одной из базовых топологий и использующих для передачи данных один из протоколов канального уровня. Внутри такой сети доставка данных обеспечивается канальным уровнем, а доставкой данных между сетями занимается сетевой уровень.

Сети соединяются между собой специальными устройствами - маршрутизаторами, которые собирают информацию о топологии межсетевых соединений и на её основе пересылают пакеты сетевого уровня в сеть назначения. Для передачи сообщения от отправителя, находящегося водной сети, к получателю в другой сети, необходимо совершить ряд транзитных передач между сетями, каждый раз выбирая подходящий маршрут – последовательность маршрутизаторов, через которые проходит пакет

Сообщения сетевого уровня формируются транспортным уровнем и в зависимости от используемого транспортного протокола называются пакетами (для протокола TCP) или дейтаграммами (для протокола UDP). Далее, отвлекаясь от способа доставки, для определенности будем называть единицу данных сетевого уровня пакетом.

**Маршрутизация** – глобальный, охватывающий всю сеть, процесс определения всего пути, который проходит пакет от отправителя до получателя. Например, при вождении автомобиля маршрутизация – это процесс прокладки маршрута в виде последовательности улиц и перекрестков.

**Продвижение данных**– это локальные действия конкретного маршрутизатора по перемещению дейтаграммы из входного порта в выходной порт. При вождении автомобиля продвижение данных соответствует процессу преодоления одного перекрестка: автомобиль подъезжает к перекрестку, водитель определяет направление до следующего перекрестка на своем пути и далее двигается к следующему перекрестку.

В современной архитектуре сети Интернет применяются все способы передачи данных, рассмотренные ранее - дейтаграммный, с установлением логического соединения и с применением виртуального канала:

* в качестве основного протокола для продвиженияпакетов используется дейтаграммный протокол IP, не дающий гарантий по доставке пакетов,сохранению исходного порядка следования пакетов, а также сохранению временных интервалов между пакетами;
* обеспечением надежной доставки данных между конечными узлами этой сети занимается протокол TCP, устанавливающий логические соединения без фиксации маршрута;
* в состав Интернета входят множество сетей ATM и FrameRelay, поддерживающих виртуальные каналы.

ATM –это сетевая высокопроизводительная технология канального уровня, которая используется в магистральных сетях и является предшественником технологии Ethernet,обеспечивая скорость передачи до 620 Мбит/с. Frame relay – протокол канального уровня, обеспечивающий скорость передачи данных до 34 Мбит/с.

## 3.1Адресация сетевого уровня

Основные сведения по адресации сетевых узлов были изложены ранее в разделе 1.6. Здесь эти вопросы будут рассмотрены более подробно.

На сетевом уровне для адресации используются IP-адреса, причем эти адреса связаны не с узлами сети, а с их сетевыми интерфейсами. В качестве сетевого интерфейса у компьютера используется сетевая карта, у маршрутизатора – порты, поэтому у одного сетевого устройства может быть несколько IP-адресов.

IP-адрес состоит из четырех байтов, каждый из которых записывается в виде десятичного целого числа. IP-адрес содержит адрес сети и адрес узла в пределах сети, причемразделение этих адресов может быть фиксированным или гибким. В первом случае для маршрутизации используется понятие класса адреса, когда граница между адресом сети и адресом узла всегда фиксируется на границе байтов.Всего существует 5 классов IP-адресов (A,B,C,D,E), но основными являются классы A, B и C. Класс определяется значениями первых четырех битов первого байта адреса. В таблице 3.1 приведены основные сведения по каждому классу.

Таблица 3.1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Класс | Первые биты | Диапазон адресов | Максимальное число узлов | Примечание |
| A | 0 | 1.0.0.0-127.255.255.255 | 224 = 16 777216 | 1 байт – номер сети,  3 байта – номер узла |
| B | 10 | 128.0.0.0-191.255.255.255 | 216 = 65536 | 2 байта – номер сети,  2 байта – номер узла |
| C | 110 | 192.0.0.0 - 223.255.255.255 | 28 = 256 | 3 байт – номер сети,  1 байта – номер узла |
| D | 1110 | 224.0.0.0 - 239.255.225.255 | особый групповой адрес | деления на номер сети и номер узла нет |
| E | 1111 | 240.0.0.0 - 255.255.255.255 | зарезервирован | в настоящий момент не используется |

Достоинством классовой адресации является простота, а недостатком – неэффективное использование ограниченных диапазонов адресов. Например, в Интернете может быть всего 126 сетей класса А, 65536 сетей класса В и 1 6777216 сетей класса С. Поэтому в настоящее время классовая адресация не используется и заменяется на бесклассовую адресацию, в которой граница между адресом сети и адресом узла является плавающей и задается с помощью маски сети. Такая адресация дает возможность более эффективно использовать диапазоны адресов путем создания подсетей.

Например, сеть класса С теоретически может содержать до 256 узлов. Но достаточно часто в небольшом офисе или учебном классе необходимо создать сеть всего из 10-12 компьютеров, для чего вполне достаточно выделить в IP-адресе 28 битов для адреса сети и 4 бита для адреса узла. При этом в одной организации в пределах выделенного ей диапазона IP-адресов можно создать несколько таких сетей с учетом потребностей каждого подразделения.

**Маска сети** – это 32-битное число, выделяющее в IP-адресе адрес сети и адрес узла: биты маски, соответствующие адресу сети, установлены в 1, остальные биты соответствуют адресу узла в этой сети и имеют значение 0. Все узлы одной сети должны использовать одну маску и иметь одинаковый адрес сети.

Возможны две формы записи маски:

* в виде 4 байтов аналогично IP-адресу:  
  *IP-адрес сети: 170.160.0.0;  
   маска подсети: 255.240.0.0;*
* указать количество начальных единичных битов в маске сети сразу после адреса сети через символ ”/” :  
  *IP-адрес сети: 170.160.0.0/12*

Алгоритм маршрутизации, использующий маски подсети, называется CIDR (Classless InterDomain Routing) – бесклассовая маршрутизация. На рис.3.1 показан пример определения адреса сети и адреса узла для компьютера с IP-адресом 170.170.170.170/12.

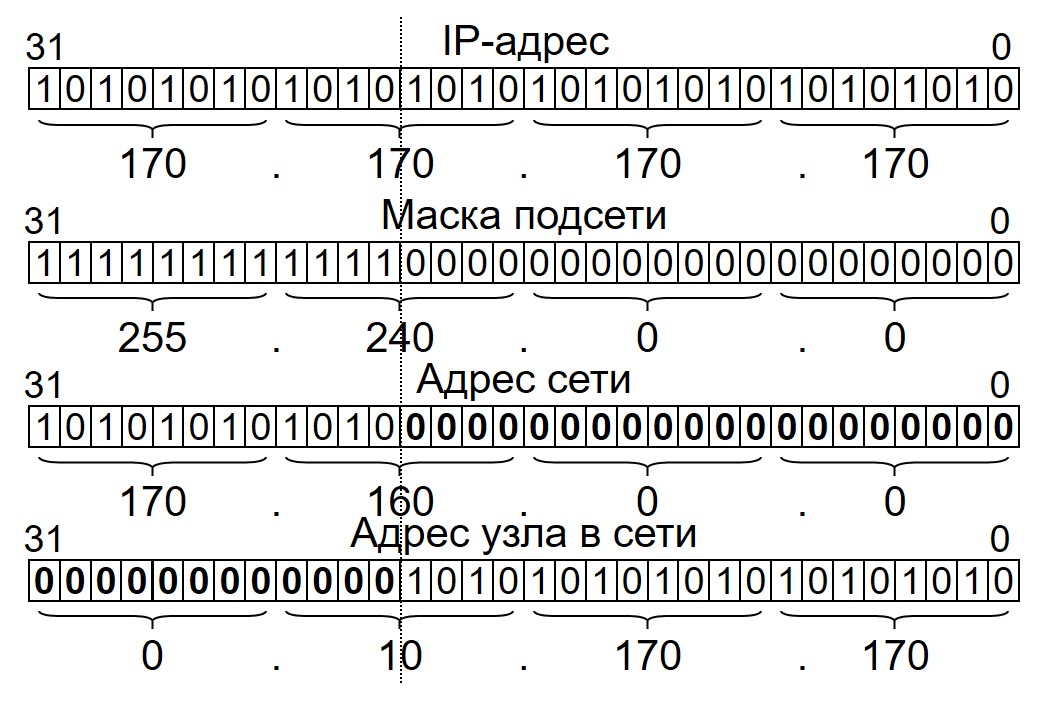


Рис. 3.1

Сетевые адреса могут быть внешними, локальными и специальными. **Внешние IP-адреса** являются уникальными в пределах глобальной сети Интернет и выделяются международной администрацией адресного пространства Интернет (IANA) в виде диапазонов класса А (/8) региональным регистраторам, которые, в свою очередь, выделяют более мелкие диапазоны адресов интернет-провайдерам. Пользователи Интернета получают внешние IP-адреса от своих провайдеров.

**Локальные IP-адреса**должны быть уникальными в пределах каждой локальной сети.Уникальность адресов обеспечивается администратором сети, который назначает IP-адреса из резервированных для таких сетей блоков адресов (10.0.0.0/8, 172.16.0.0/16 или 192.168.0.0/24).

В разных локальных сетях локальные адреса часто повторяются, поэтому при направлении любого запросаотлокального компьютераво внешнюю сеть Интернет возникает проблема с получением ответа, так как обращение из Интернета по локальному адресу невозможно из-за неоднозначной идентификации отправителя запроса. Для ее решения достаточно одного маршрутизатора, имеющего общий внешний адрес и функцию преобразования сетевых адресов NAT.

Каждый пакет сетевого уровня, направляемый из компьютера локальной сети, содержит IP-адреса получателя и отправителя. Маршрутизатор выделяет из пакета эти адреса и, если адрес получателя является внешним, выполняет подмену адреса отправителя на свой внешний адрес, доступный из Интернета.Локальный адрес отправителя и номер портасохраняются в отдельной таблице маршрутизатора, чтобы различать ответные пакеты, адресованные разным локальным компьютерам. При получении ответа от внешнего компьютера выполняется обратное действие – в пакет вместо своего внешнего адреса маршрутизатор подставляет сохраненный локальный адрес и направляет этот пакет на нужный порт. После того, как клиент и сервер закончат обмениваться пакетами, записи в таблице удаляются.

**Специальные IP-адреса** предназначены для широковещательных и групповых рассылок, а также для отладки сетевых приложений. В протоколе IP существует несколько соглашений по поводу возможных значений таких IP-адресов.

1. Номера сетей и узлов не могут состоять из одних нулей или единиц, поэтому максимальное расчетное число узлов надо уменьшать на два. Например, для сети с маской /24– это 254 узла.

2. Адрес, у которого *все биты равны 1*, называется ограниченным широковещательным адресом (limited broadcast).Пакет с таким адресом должен рассылаться всем узлам, находящимся в одной сети с отправителем. Ограниченность означает, что пакет не выйдет за пределы данной сети.

3. Адрес, у которого *все биты адреса узла равны 1*, называется широковещательным адресом (broadcast). Пакет с таким адресом будет рассылаться всем узлам сети, номер которой указан в адресе. Например, пакет с адресом 192.190.22.255/24 будет направлен всем узлам сети 192.190.22.0,узел не может иметь такой адрес.

4. Адрес, у которого *все биты адреса узла равны 0*, используется как адрес сети, узел не может иметь такой адрес.

5. Адрес, у которого *первый байт адреса равен 127*, является внутренним адресом стека протоколов TCP/IP компьютера или маршрутизатора (loopback, петля) и используется для тестирования сетевых приложений или для организации клиент-серверного взаимодействия на одном компьютере.

6. Групповые адреса (см. табл.3.1) предназначены для передачи пакетов всем узлам, имеющих этот адрес (multicast). Один узел может иметь индивидуальный и несколько групповых IP-адресов.

Деление IP-адресов на внешние и локальные связано с тем, что адрес размером 4 байта, используемый протоколом IPv4, ограничивает количество устройств, подключенных к сети Интернет, значением 232= 4294967296 единиц. Однако по данным на январь 2021 года число пользователей Интернет составляет более 4,66 миллиардов человек, а число только мобильных подключений составляет более 8,02 миллиардов [10].По данным международной компании IDC к 2025 году в сети будет более 152 миллиардов устройств Интернета вещей [11].

Для решения этой проблемы был разработан протокол IPv6, в котором для IP-адреса выделено 16 байтов. Содержимое каждого байта IPv6-адреса записывается двухбайтовыми сегментами в шестнадцатиричном коде, в качестве разделителя сегментов применяется символ двоеточие (:). Для сокращения записи IPv6-адресов используются следующие соглашения:

* можно пропускать все начальные нули в сегменте.Например, 00BC можно записывать как BC, 05АD – как 5AD;
* можно использовать двойное двоеточие (::), которое заменяет любую смежную строку из одного или нескольких сегментов, состоящих из нулей. Двойное двоеточие (::) может использоваться в адресе только один раз.

Например, адрес fe80:0000:0000:0000:f482:5fec:0acc:2ffa можно записать в виде fe80::f482:5fec:acc:2ffa

В настоящее время основную долю в мировом трафикезанимает IPv4, так каксогласно статистике Google на январь 2022 года доля IPv6 в сетевом трафике составляла около 40 %, в России эта доля была около 7,5 % [12]. Поэтому далее под термином «IP» будем понимать протокол IPv4.

## 3.2 Связь физических, сетевых и доменных адресов

Кадры, передаваемые между узлами сети на канальном уровне, несут информацию только о физических адресах получателя и отправителя. На сетевом уровне носителями данных являются пакеты (дейтаграммы), содержащие IP-адреса получателя и отправителя. На прикладном уровне для указания получателя данных пользователи чаще всего используют доменные адреса.

Однозначной связи между физическими, сетевыми и доменными адресами не существует. Поэтому единственным способом для преобразования доменного имени в сетевой адрес, а сетевого адреса в физический адрес, является применение ассоциативных таблиц, каждая запись в которых содержит непосредственное описание соответствующей связи.

### 3.2.1 Протокол ARP

Физический (MAC) адрес необходим для передачи информации в пределах локальной сети стандартной топологии: шина, кольцо или звезда. IP-адрес узла назначается администратором и с MAC-адресом прямо не связан, поэтомудля определения MAC-адреса (Ethernet-адреса) по IP-адресу на канальном уровне используется протокол разрешения адресов ARP (Address Resolution Protocol). Также существует протокол RARP (Reverse ARP), реализующий обратную задачу - нахождение IP адреса по физическому адресу (рис. 3.2).

ARP

RARP

IP

Ethernet

Рис. 3.2

Протокол ARP работает на основе широковещательных запросов, поэтому для уменьшения трафика, генерируемого такими запросами, каждый сетевой интерфейс имеет встроенную ARP-таблицу, в которой хранятся IP-адреса всех узлов локальной сети и соответствующие им МАС-адреса (табл. 3.2).

Таблица 3.2

|  |  |  |
| --- | --- | --- |
| Адрес IP | Физический адрес (MAC) | Тип связи |
| 195.62.2.1 | 00-02-16-09-fa-40 | динамический |
| 195.62.2.11 | 00-90-27-a1-36-d0 | динамический |
| 195.62.2.15 | 00-08-c7-91-c1-5a | динамический |
| 195.62.2.11 | 00-17-d1-64-f7-a8 | статический |

Записи в таблицу могут добавляться администратором сети или протоколом ARP. В первом случае записи называются статическими и хранятся в таблице до следующей перезагрузки устройства, во втором случае – динамическими и хранятся в течение определенного интервала времени.

При поступлении IP-пакетав локальную сетьполучателя запускается протокол ARP соответствующего порта маршрутизатора и в ARP-таблице проводится поиск IP-адреса назначения полученного пакета. При положительном результате поиска пакет направляется узлу с найденным MAC-адресом, иначе генерируется широковещательный ARP-запрос “*Какой физический адрес имеет сетевой интерфейс с адресом IPD ?*”, где IPD – это IP-адрес получателя пакета. Этот запрос получают все узлы локальной сети назначения и один из них, имеющий адрес IPD , отвечает отправителю запроса, сообщая свой MAC-адрес. Этот MAC-адрес протокол ARP маршрутизатора добавляет в свою ARP-таблицу, после чего пакет направляется получателю.

Когда компьютер с некоторым IP-адресом решает отправить кадр с данными другому компьютеру, он выясняет, принадлежит ли адресат собственной локальной сети, накладывая на IP-адрес получателя маскусети. Если полученныйадрес совпадает с адресом сети отправителя, то оба находятся в одной локальной сети и MAC-адрес получателя берется из ARP-таблицы отправителя. При отсутствии в таблице IP-адреса получателя протокол ARPотправителя посылает широковещательный запрос всем узлам сети с просьбой откликнуться и сообщить свой MAC-адрес компьютер, который имеетзаданныйIP-адрес. После получения ответа отправительдобавляет новую запись в свою ARP-таблицу и отправляет кадр получателю.

На рис.3.3 показана структура кадра протокола ARP. Размер кадра составляет 28 байтов, поля кадра имеют следующие значения:

* тип оборудования – номер канального протокола передачи данных (для Ethernet равен 0x0001);
* тип протокола – код сетевого протокола, для IPv4 равен 0x0800;

****

Рис.3.3

* HA-Len – длина физического адреса в байтах, для Ethernet равна 0x06;
* PA-Len – длина логического адреса в байтах, для адреса IPv4 равна (0x04);
* код операции –для ARP-запросаравен 0x0001, дляARP- ответа – 0x0002;
* MAC-адрес и IP-адрес отправителя;
* MAC-адрес (при запросе заполняется нулями) и IP-адрес адресата.

Для просмотра ARP-таблицы можно использовать команду arp –a, которая присутствует в ОС Linux и Windows.

### 3.2.2 Протокол DNS

Доменная система имен, используемая стекомTCP/IP, имеет иерархическую древовидную структуру. Под доменом имен понимается совокупность имен, у которых несколько старших составных частей совпадают. Например, имена *yandex.ru, kp.ru, lenta.ru*входят в домен первого (верхнего) уровня *.ru*, а имена *yandex, kp*и *lenta*являются доменами второго уровня, в каждом из которых могут создаваться домены третьего уровня (например, *pogoda.yandex.ru*) и т.д.. Разделение доменных имен на части позволяет разделить административную ответственность за назначение уникальных имен между различными организациями в пределах одного уровня иерархии.

В каждом домене имен сети Интернет имеется выделенныйDNS-сервер, на котором хранятся таблицы соответствия доменных и сетевых имен всех узлов домена, а также ссылки на адреса DNS-серверов родительского домена и всех поддоменов.Таким образом данные DNSхранятся в виде набора таблиц,распределенных по большому числу узлов Интернета.Например, DNS–сервер домена *nstu.ru*хранит таблицу, в которой зарегистрированы имена *ami.nstu.ru, fb.nstu.ru, cs.nstu.ru*и т.д., а сервер поддомена *ami.nstu.ru* хранит таблицу с именами*fpm2.ami.nstu.ru, saturn.ami.nstu.ru, moodle.ami.nstu.ru*и т.д.

Преобразование доменного имени в сетевой адрес реализуется на основе прикладного протокола DNS, программное обеспечение которого реализовано в технологии «клиент - сервер». DNS-клиенты входят в состав сетевого ПО всех операционных систем и предназначены для формирования запросов к DNS-серверам.

Для ускорения поиска IP-адресов все DNS-серверы используют кэширование проходящих через них ответов. Записи в DNS-кэше могут быть статическими и динамическими, динамические записи сохраняются в течение определенного срока, который может составлять до нескольких дней.DNS-клиенты также записывают все результаты своих запросов в собственный DNS-кэш. Просмотреть содержимое кэша можно командой **ipconfig /displaydns**(рис. 3.4),очистить кэш – командой **ipconfig /flushdns**.

Поиск IP-адреса в системе DNS проводится в следующем порядке:

* собственный DNS-кэш сетевого узла;
* DNS-сервер локальной сети, адрес которого указан в параметрах настройки сетевого интерфейса;
* DNS-сервер родительского домена и так далее до домена верхнего уровня.
* В случае отрицательного результата поиска выводится сообщение об ошибке.

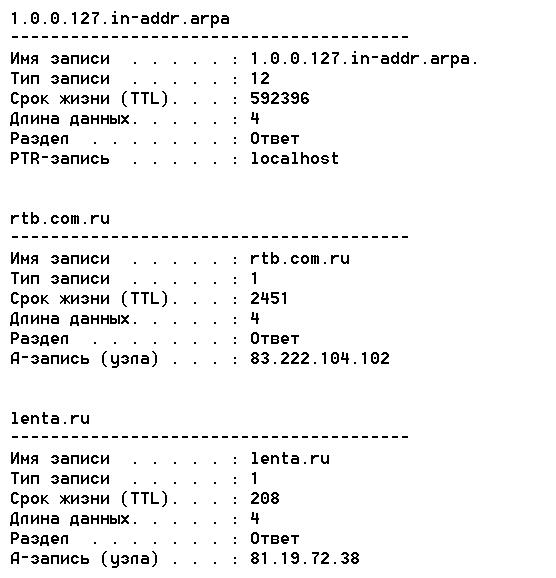


Рис. 3.4

## 3.3Протоколы сетевого уровня

Сетевой уровень стека протоколов TCP/IP представлен тремя компонентами (рис.3.5):

* протокол IP, определяющий адресацию сетевого уровня, формат полей пакета и действиянад пакетом, выполняемые маршрутизаторами и конечными узлами;
* протоколы маршрутизации, выбирающие маршрут следования пакета от отправителя к получателю и реализующие функцию продвижения пакета.Примерами протоколов маршрутизации являются RIP, OSPF, BGP и др.
* протокол ICMP, формирующий сообщения об ошибках, возникающих на пути продвижения пакетов и отвечающий на запросы служебной информации сетевого уровня.

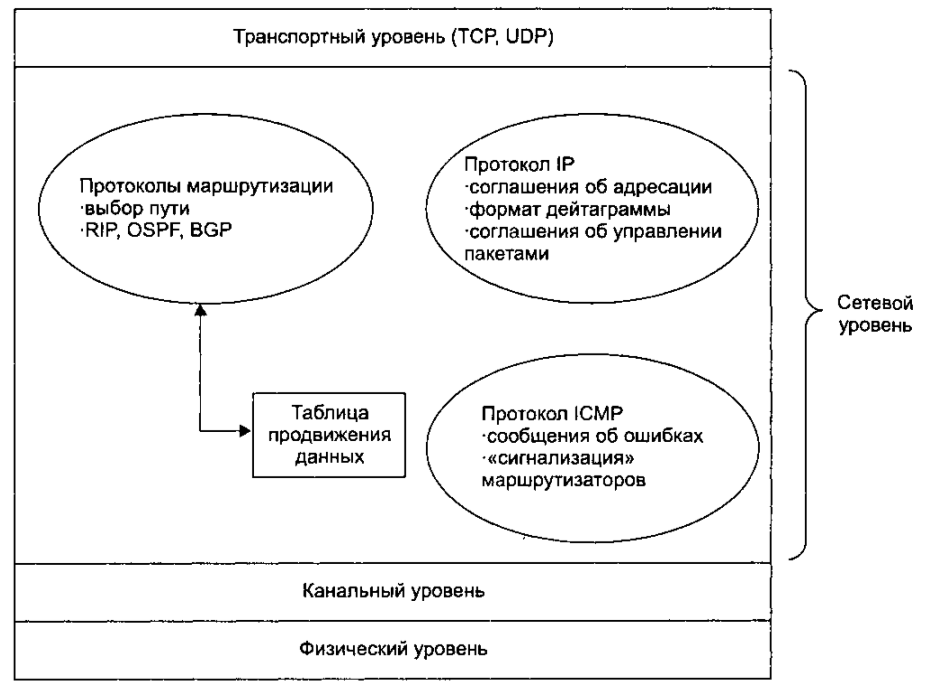


Рис. 3.5

Протокол IP использует дейтаграммный способ доставки данных и не дает гарантии их надежной доставки получателю. Здесь следует различать термины «дейтаграммный способ доставки» и «дейтаграмма». Дейтаграммный способ доставки основан на том, что при любой ошибке пакет отбрасывается, а отправителю в большинстве случаев посылается соответствующее ICMP-сообщение. Обеспечение надежности доставки при этом возлагается на более высокий уровень – транспортный (протокол TCP) или прикладной. Дейтаграмма – это данные, которые передаются от транспортного уровня на сетевой уровень протоколом UDP и инкапсулируются в IP-пакеты (см. раздел 1.8.1).

Строго говоря, в описании модели OSI протокольная единица данных сетевого уровня называется пакетом, но в некоторых источниках эта единица называется дейтаграммой. Нам важно понять структуру данных, передаваемых протоколом IP, независимо от того, как мы эти данные назовем. Формат пакета протокола IP приведен на рис. 3.6. Ниже описано назначение каждого поля.

*Номер версии.* Поле определяет номер версии протокола IP. По этому номеру маршрутизатор может определить, как интерпретировать остальные поля IP-дейтаграммы. На рисунке показан формат пакета версии IPv4.

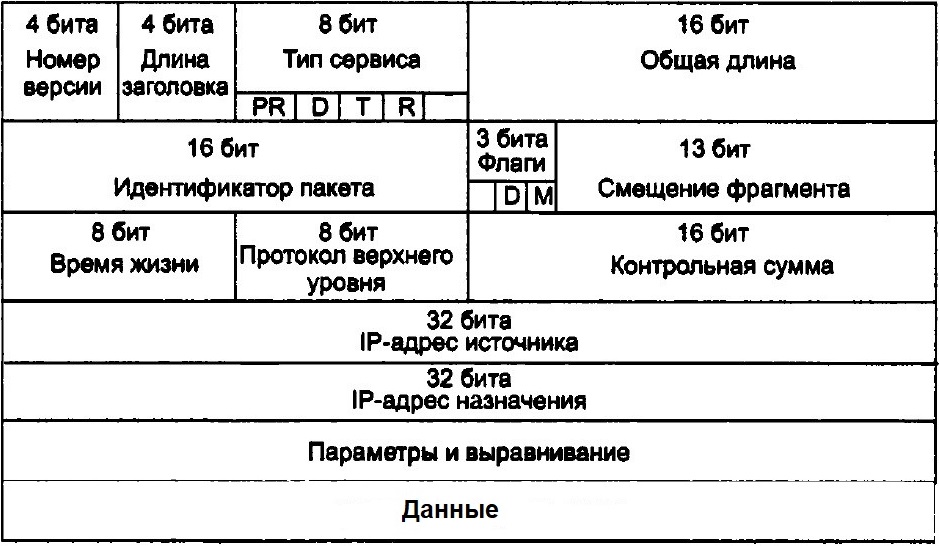


Рис.3.6

*Длина заголовка*. Пакет может содержать разное количество необязательных полей параметров, включаемых в заголовок, поэтому эти четыре бита необходимы для того, чтобы определить, где заканчивается заголовок. В большинстве IP-пакетов не содержатся поля параметров, поэтому обычно заголовок занимает 20-байтов (пять слов по 32 разряда).

*Тип сервиса*. Это поле называется «байт дифференцированного обслуживания» и указывает тип пакета (например, стандартный трафик, IP-телефония, видеотрафик и т.д.). Изменением значения этого поля можно управлять приоритетом обработки пакетов в маршрутизаторах.

*Общая длина*. Это полная длина IP-пакета в байтах (заголовок и данные). Размер поля равен 16 битов, поэтому теоретически максимальный размер пакета может составлять 65 535 байт, однако размер дейтаграмм редко превосходит 1500 байт.

*Идентификатор пакета*.Уникальный номер пакета, используемый для распознавания пакетов при сборке в случае их фрагментации; все фрагменты одного пакета имеют одинаковое значение этого поля;

*Флаги*содержат признаки, связанные с фрагментацией;

*Смещение фрагмента.* Задает смещение поля данных этого фрагментаотносительно начала поля данных исходного пакета;

*Время жизни (Time То Live, TTL).*Задает максимальное число транзитных узлов, которое разрешено пройти пакету; при прохождении каждого узла значение TTL уменьшается на единицу и при достижении нулевого значения пакет уничтожается. Максимальное значение TTL зависит от ОС (обычно для FreeBSD–54, для Linux–64, для Windows–128, для Cisco–255). Для Windows реальное значение TTL при отправке пакета определяется ключом реестра DefaultTTL, значение которого можно определить пингованием адреса 127.0.0.1. При отправке пакета в другую сеть он уходит со значением TTL, которое установлено на шлюзе (маршрутизаторе) локальной сети.

*Протокол верхнего уровня*. Это поле используется только тогда, когда IP-пакет достигает конечного адресата. Значение поля определяет протокол транспортного уровня, которому следует передать данные из пакета. Например, значение 6 означает, что пакетнадо передать протоколу TCP, 17 — протоколу UDP, 1 - протоколу ICMP. Номер протокола в IP-пакете связывает сетевой и транспортный уровни.

*Контрольная сумма заголовка.* Контрольная сумма заголовка используется для обнаружения ошибок в заголовках полученных IP-пакетов. Пакеты, в которых обнаружены ошибки, отбрасываются. Контрольная сумма вычисляется на каждом маршрутизаторе заново и снова сохраняется в поле заголовка, так как каждый раз изменяется TTL и могут измениться значения поля параметров.

*IP-адреса отправителя и получателя.* Эти поля содержат 32-разрядные IP-адреса отправителя и конечного получателя IP-дейтаграммы. Следует упомянуть широковещательный IP-адрес, когда хост передает пакет с адресом получателя 255.255.*255.255 и сообщение доставляется всем узлам той же сети.*

*Параметры.* Поле параметров позволяет расширить IP-заголовок. Параметры используются редко и поэтому не включаются в заголовок дейтаграммы для сни-жения накладных расходов.

*Данные*. В большинстве случаев поле данных IP-пакета содержит сегмент транспортного уровняTCP или дейтаграмму UDP, которые необходимо доставить адресату. Однако поле данных может содержать и другие типы данных, например, сообщения протокола ICMP.

**Протокол межсетевых управляющих сообщений ICMP** предназначен для формирования и передачи сообщений о состоянии узла назначения IP-пакета и ошибках, возникающих во время прохождения IP-пакета по сети. Маршрутизатор, который обнаруживает ошибку, выполняет следующие действия:

* формирует ICMP-пакет о причине ошибки;
* формирует новый IP-пакет и вкладывает в него свой ICMP-пакет;
* направляет сформированный IP-пакет отправителю пакета, в котором произошла ошибка;
* ошибочный IP-пакет отбрасывается.

Структура ICMP-пакета приведена на рис. 3.7 и содержит заголовок и поля данных. Заголовок включает обязательную и необязательную части по 4 байта каждая.



Рис. 3.7

Обязательная часть имеет три поля:

* тип – указывает основную причину формирования сообщения;
* код – уточняет причину формирования сообщения;
* контрольная сумма сообщения.

На рис. 3.8 приведены основные типы ICMP-сообщений. Наиболее часто передаются следующие типы:

* *Запрос/ответ эхо - повтора*. Запрос эхо - повтора может быть передан программному обеспечению протокола ICMP любого компьютера. В ответ на запрос эхо – повтора программное обеспечение ICMP должно послать ответ эхо – повтора ICMP, который содержит те же данные, что и запрос. Этот тип сообщений используется программами **ping** и **tracert.**
* *Узел назначения недостижим*. Обнаружив, что пакет не может быть доставлен в место назначения, маршрутизатор отправляет узлу, на котором он был создан, сообщение о том, что получатель недоступен. В сообщении указано, является ли недоступным конкретный узел назначения или недоступна сеть, к которой подключен этот узел.
* *Истечение времени пакета*. Сообщение формируется при выполнении условия TTL=0.



Рис. 3.8

Интерпретация необязательной части заголовка и поля данных зависят от типа сообщения. Например, для сообщений типа 8 (эхо-запрос) и 0 (эхо-ответ) необязательная часть заголовка может содержать поля «Идентификатор» и «Номер», которые используются отправителем для проверки соответствия между запросом и ответом, а поле данных для сообщений типа 3 (адресат недоступен), 5 (перенаправление маршрута) и 11 (истечение TTL) содержит заголовок и первые 8 байтов поля данных IP-пакета, вызвавшего ошибку, чтобы отправитель мог идентифицировать этот пакет. Поле данных имеет переменную длину и содержит данные, которые необходимо вернуть отправителю.

## 3.4Основы IP-маршрутизации

**Маршрутизатор** – это многопортовое сетевое коммуникационное устройство, предназначенное для формирования маршрута передачи данных от отправителя к получателю и продвижения IP-пакетов между разными сетями. Каждый порт маршрутизатора подключается к отдельной сети, имея собственные сетевой адрес и локальный адрес,поэтому его можно рассматривать как совокупность узлов, каждый из которых входит в свою сеть. Как единое устройство маршрутизатор не имеет выделенного адреса (сетевого или локального) для продвижения пакетов, но может иметь отдельный порт с собственным IP-адресом для удаленного управления.

Все маршрутизаторы периодически обмениваются с соседними маршрутизаторами информацией о доступных им сетях, собирая таким образом информацию о текущей конфигурации межсетевых соединений. Реализация маршрутизаторов может быть аппаратной или программной. В конечных устройствах применяется, как правило, программная реализация.

Для продвижения пакетов используется таблица маршрутизации, в которой хранятся сведения о доступных сетях:

* *сетевой адрес* – это IP-адрес назначения,в качестве которого могут быть указаны адрес маршрута по умолчанию, адрес сети или адрес конкретного узла, к которомунеобходимо зафиксировать маршрут.
* *маска сети назначения*;
* *адрес шлюза* – это IP-адрес следующего маршрутизатора на маршруте продвижения пакета;
* *метрика*, которая характеризует критерий выбора маршрута в случае нескольких альтернативных вариантов; в качестве метрики могут использоваться пропускная способность, текущая загруженность маршрута или длина маршрута, определяемая количеством хопов – пройденных на маршруте промежуточных маршрутизаторов для достижения сети назначения.
* *IP -адрес выходного порта* (интерфейса), через который надо передать пакет.

На рис.3.9 приведен фрагмент таблицы маршрутизации программного маршрутизатора Windows, который можно просмотреть командой **route print** .

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Сетевой адрес | Маска | Адрес шлюза | Интерфейс | Метрика |
| 127.0.0.0 | 255.0.0.0 | 127.0.0.1 | 127.0.0.1 | 1 |
| 0.0.0.0 | 0.0.0.0 | 198.21.17.7 | 198.21.17.5 | 1 |
| 129.13.0.0 | 255.255.0.0 | 198.21.17.6 | 198.21.17.5 | 2 |
| 198.21.17.0 | 255.255.255.0 | 198.21.17.5 | 198.21.17.5 | 1 |
| 198.21.17.5 | 255.255.255.255 | 127.0.0.1 | 127.0.0.1 | 1 |
| 198.21.17.255 | 255.255.255.255 | 198.21.17.5 | 198.21.17.5 | 1 |
| 213.34.12.0 | 255.255.255.0 | 213.34.12.3 | 213.34.12.3 | 1 |
| 213.34.12.3 | 255.255.255.255 | 127.0.0.1 | 127.0.0.1 | 1 |
| 255.255.255.255 | 255.255.255.255 | 198.21.17.6 | 198.21.17.6 | 1 |

Рис. 3.9

Запись, у которой метрика равна 1, соответствует сети, непосредственно подключенной к маршрутизатору. Такой признак говорит о том, что пакет уже достиг своей сети. Некоторые маршрутизаторы для непосредственно подключенных сетей в поле метрики выводят значение On-link.

Алгоритм просмотра таблицы маршрутизации следующий:

1. При поступлении на любой из портов IP-пакета маршрутизатор извлекает из него IP-адрес получателя (IPD).
2. Проводится поиск конкретного (специфического) маршрута для адреса IPD .Для каждой записи таблицы, имеющей значение маски 255.255.255.255, извлекается адрес назначения и сравнивается с адресом IPD . Если совпадение произошло, то из этой строки извлекаются адрес шлюза и адрес выходного интерфейса. На этом просмотр таблицы заканчивается и пакет направляется на выходной интерфейс.
3. Если совпадения не произошло, то протокол IP переходит ко второй фазе просмотра - поиску маршрута к сети назначения. Из адреса IPD с помощью маски выделяется адрес сети и таблица снова просматривается по строкам для поиска совпадения адреса сети в какой-либо строке с адресом сети из пакета. При совпадении из строки таблицы извлекаются адрес шлюза и адрес выходного интерфейса. На этом просмотр таблицы заканчивается.
4. Если в п.2 и п.3 найдено несколько записей, удовлетворяющих условиям поиска, то маршрут выбирается на основе лучшей метрики.
5. Если адрес назначения не найден в п.2 и п.3, то выбирается маршрут по умолчанию, которыйзадается в строке, имеющей маску 0.0.0.0, так как наложение такой маски на любой IP-адрес даст результат 0.0.0.0, что совпадает с адресом назначения, указанным в этой строке таблицы. Если маршрут по умолчанию не задан, то полученный пакет отбрасывается и просмотр таблицы на этом заканчивается.

*Обратите внимание! В маршрутизаторе, в отличие от коммутатора, при неизвестном адресе назначения пакет отбрасывается !*

Таблицу маршрутизации программного маршрутизатора Linux можно просмотреть командами **ip route** или **route** (рис. 3.10).

[kvg@students ~]$ route

Kernel IP routing table

Destination Gateway Genmask Flags Metric Ref Use Iface

default gw-130.ami.nstu 0.0.0.0 UG 100 0 0 eth0

192.168.122.0 0.0.0.0 255.255.255.0 U 0 0 0 virbr0

217.71.130.128 0.0.0.0 255.255.255.128 U 100 0 0 eth0

Рис. 3.10

Назначение столбцов здесь следующее:

* Destination – адрес назначения,
* Gateway – адрес шлюза;
* Genmask – маска сети;
* Flags – флаги записей (U–маршрут активен и работоспособен, H–признак маршрута к конкретному узлу; G–маршрут проходит через промежуточный маршрутизатор (шлюз), отсутствие этого флага означает непосредственно подключенную сеть);
* Metric – метрика маршрута;
* Ref – число ссылок на маршрут при продвижении пакетов;
* Use – число байтов, переданных по маршруту;
* Iface – имя выходного интерфейса

Существует три основных источника записей в таблице маршрутизации:

* программное обеспечение стека TCP/IP, которое при инициализации маршрутизатора заносит в таблицу записи о непосредственно подключенных сетях и маршрутах по умолчанию;
* администратор сети, добавляющий записи таблицы с помощью команд ОС (например, route) или команд маршрутизатора. Заданные таким образом записи всегда являются статическими и не имеют срока жизни. Эти записи могут быть постоянными (сохраняются при перезагрузке маршрутизатора) или временными (хранятся только до выключения устройства);
* протоколы маршрутизации, такие как RIP или OSPF. Эти записи всегда являются динамическими, то есть имеют ограниченный срок жизни.

Одним из протоколов маршрутизации, используемых в небольших сетях, является протокол RIP.В современных маршрутизаторах, работающих с масками сетей, используется версия протокола RIPv2.

Метрика, используемая протоколом RIP должна обладать свойством аддитивности, то есть метрика составного пути должна быть равна сумме метрик составляющих этого пути. В большинстве реализаций RIP в качестве метрики используется количество хопов.

Рассмотрим алгоритм построения таблицы маршрутизации с помощью RIP.

1. Создание минимальной таблицы. В исходном состоянии программное обеспечение стека TCP/ IP автоматически создает минимальную таблицу маршрутизации, в которой учитываются только непосредственно подключенные сети.

2. Рассылка минимальной таблицы соседним маршрутизаторам. После инициализации каждый маршрутизатор начинает посылать своим соседям сообщения протокола RIP, в которых содержится его минимальная таблица. RIP-сообщения передаются дейтаграммами протокола UDP и для каждой сети включают два параметра: ее IP-адрес и расстояние до нее от передающего сообщение маршрутизатора.

3. Получение RIP-сообщений от соседей и обработка полученной информации. После получения RIP-сообщений от соседей маршрутизатор наращивает каждое полученное поле метрики на единицу, запоминает, через какой порт и от какого маршрутизатора получена новая информация. Запись о новой сети добавляется в таблицу, если адрес этой сети отсутствует в таблице, или если значение метрики новой записи меньше, чем у существующей (старая запись при этом удаляется). В результате в таблице маршрутизации о каждой сети остается только одна запись.

4. Рассылка новой таблицы соседям. Каждый маршрутизатор отсылает новое RIP-сообщение всем соседям. В этом сообщении он помещает данные обо всех известных ему сетях: как непосредственно подключенных, так и удаленных, о которых маршрутизатор узнал из RIP-сообщений.

5. Получение RIP-сообщений от соседей и обработка полученной информации. Этап 5 повторяет этап 3 — маршрутизаторы принимают RIP-сообщения, обрабатывают содержащуюся в них информацию и на ее основании корректируют свои таблицы маршрутизации.

После нескольких итераций рассылки и обработки RIP-сообщений таблицы всех маршрутизаторов сети устанавливаются в такое состояние, когда все сети достижимы из любой сети с помощью некоторого рационального маршрута.

При изменении конфигурации сети, связанной с появлением новых маршрутизаторов или отключением старых, необходимо внести соответствующие изменения в таблицы маршрутизации. Новые маршрутизаторы после подключения начинают рассылку RIP-сообщений своим соседям и постепенно эта информация становится известна всем маршрутизаторам сети. При отключении старых маршрутизаторов необходимо разослать всем сообщения о том, что некоторые маршруты стали недействительными, для чего используются два способа: истечение времени жизни маршрута или указание специального (бесконечного) расстояния до сети, ставшей недоступной.

*Механизм истечения времени жизни маршрута* (TTL) основан на том, что каждая запись таблицы маршрутизации, полученная по протоколу RIP, имеет определенное время жизни. При поступлении очередного RIP-сообщения, подтверждающего корректность данной записи, таймер TTL устанавливается в исходное состояние, а затем из него каждую секунду вычитается единица. Маршрут считается действительным до тех пор, пока TTL>=0.

В протоколе RIP период рассылки равен 30 сек, а в качестве тайм-аута выбрано шестикратное значение периода (180 сек).  Если маршрутизатор перестает рассылать своим соседям сообщения о сетях, которые можно через него достичь, то через 180 секунд все записи, порожденные этим маршрутизатором, у его соседей станут недействительными, а после этого процесс повторится уже для ближайших соседей — они вычеркнут подобные записи уже через 360 секунд.

*Механизм тайм-аута работает*, когда маршрутизатор не может послать соседям сообщение об отказавшем маршруте из-за собственного отказа или из-за отказа линии связи. Если отправка сообщения возможна, то маршрутизаторы RIP указывают «бесконечное» расстояние до сети, ставшей недоступной.

В протоколе RIP бесконечным условно считается расстояние в 16 хопов. Получив сообщение, в котором расстояние до некоторой сети равно 15, маршрутизатор должен проверить, исходит ли эта «плохая» информация о сети от того же маршрутизатора, сообщение которого послужило в свое время основанием для записи о данной сети в таблице маршрутизации. Если это тот маршрутизатор, то информация считается достоверной и маршрут помечается как недоступный.

По этой причине применение протокола RIP возможно только в сетях, имеющих не более 15 маршрутизаторов. Для более масштабных сетей нужно применять другие протоколы маршрутизации, например, OSPF.

## 3.5 Диагностика IP-сети

### 3.5.1 Команды диагностики IP-сетей

В состав каждой ОС входит набор утилит, предназначенный для диагностики оборудования сетей TCP/IP. Рассмотрим некоторые из этих утилит.

3.5.1.1 Команда: **ipconfig**

Команда **ipconfig**(в Linux – **ifconfig**) предназначена для управления сетевыми интерфейсами. Она позволяет просмотреть параметры настройки всех интерфейсов в кратком (по умолчанию) или в полном виде (ключ **/all**), просмотреть и очистить кэш DNS (**/displaydns** и **/flushdns**), обновить (**/renew**) или освободить (**/release**) IP-адрес любого интерфейса и т.д. Ниже приведен пример полного вывода информации по интерфейсам для ОС Windows и Linux.

**Z:\>ipconfig /all**

Настройка протокола IP для Windows

Имя компьютера . . . . . . . . . : pmi-srv-terminal

Основной DNS-суффикс . . . . . . : corp.nstu.ru

Тип узла. . . . . . . . . . . . . : Гибридный

IP-маршрутизация включена . . . . : Нет

WINS-прокси включен . . . . . . . : Нет

Порядок просмотра суффиксов DNS . : corp.nstu.ru ami.nstu.ru

Ethernet adapter Ethernet:

DNS-суффиксподключения . . . . . : ami.nstu.ru

Описание. . . . . . . . . . . . . : Сетевой адаптер Hyper-V (Майкрософт)

Физический адрес. . . . . . . . . : 00-15-5D-82-8D-BF

DHCP включен. . . . . . . . . . . : Да

Автонастройка включена. . . . . . : Да

IPv4-адрес. . . . . . . . . . . . : 217.71.130.188(Основной)

Маска подсети . . . . . . . . . . : 255.255.255.128

Аренда получена. . . . . . . . . . : 31 октября 2022 г. 15:25:15

Срок аренды истекает. . . . . . . . . . : 4 ноября 2022 г. 0:25:15

Основной шлюз. . . . . . . . . : 217.71.130.254

DHCP-сервер. . . . . . . . . . . : 217.71.130.130

DNS-серверы. . . . . . . . . . . : 217.71.130.130

Основной WINS-сервер. . . . . . . : 192.168.100.200

Дополнительный WINS-сервер. . . . . . : 217.71.130.160

NetBios через TCP/IP. . . . . . . . : Включен

**[kvg@students ~]$ifconfig**

eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500

inet 217.71.130.131 netmask 255.255.255.128 broadcast 217.71.130.255

inet6 2001:b08:a:1040:a9ee:5fc5:451f:5cf prefixlen 128 scopeid 0x0<global>

inet6 fe80::215:5dff:fe82:8d01 prefixlen 64 scopeid 0x20<link>

inet6 2001:b08:a:1040:215:5dff:fe82:8d01 prefixlen 64 scopeid 0x0<global>

ether 00:15:5d:82:8d:01 txqueuelen 1000 (Ethernet)

RX packets 100722090 bytes 69784724650 (64.9 GiB)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 31856075 bytes 10623260236 (9.8 GiB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536

inet 127.0.0.1 netmask 255.0.0.0

inet6 ::1 prefixlen 128 scopeid 0x10<host>

loop txqueuelen 0 (Local Loopback)

RX packets 11702641 bytes 7345008270 (6.8 GiB)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 11702641 bytes 7345008270 (6.8 GiB)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

virbr0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500

inet 192.168.122.1 netmask 255.255.255.0 broadcast 192.168.122.255

ether 52:54:00:d4:60:b6 txqueuelen 0 (Ethernet)

RX packets 0 bytes 0 (0.0 B)

RX errors 0 dropped 0 overruns 0 frame 0

TX packets 0 bytes 0 (0.0 B)

TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

Из примера следует, что:

* компьютер с именем pmi-srv-terminal работает под управлением Windows и имеет один сетевой интерфейс с IP-адресом 217.71.130.188;
* компьютер под управлением Linux имеет три сетевых интерфейса, основной из которых имеет адрес 217.71.130.131;
* оба компьютера находятся в одной сети с маской 255.255.255.128.

В Linuxназвание сетевого интерфейса состоит из его типа и порядкового номера(N). Все сетевые интерфейсы Ethernet в Linux называются ethN, начиная с *eth0*. Для каждого интерфейса указываются физические и сетевые адреса (*ether, inet, inet6*),маска сети (*netmask*), максимальный размер кадра (*MTU*), число принятых и переданных пакетов (*RXpackets, TXpackets*).

Интерфейс *lo* – это интерфейс внутреннего стека TCP/IP (петля или заглушка), предназначенный для отладки сетевого ПО и имеющий IP-адрес 127.0.0.1. Интерфейс *virbr0*является виртуальным мостом, который связывает созданные пользователем виртуальные машины с основной машиной.

3.5.1.2 Команда: **nslookup**

Утилита  **nslookup**является аналогом DNS-клиента и позволяет диагностировать проблемы, возникающие в системе DNS. По умолчанию все запросы отправляются на DNS-сервер, адрес которого задан настройками сетевого подключения (см. команду **ipconfig**).

При запуске **nslookup** без параметров утилита переходит в интерактивный режим, ожидая ввод команд пользователя. Ввод символа ”?” или команды **help**  (в Windows) отображает справку о внутренних командах и опциях программы **nslookup**. В Linux получение справки проводится с помощью команды **man**.

Ниже приведен результат выполнения команды **nslookup**в Windows и Linux.

**C:\>nslookup yandex.ru**

Server: UnKnown

Address: 192.168.1.1

Name: yandex.ru

Addresses: 77.88.55.70, 77.88.55.66, 5.255.255.80, 5.255.255.70

**[kvg@students ~]$nslookup yandex.ru**

Server: 217.71.130.130

Address: 217.71.130.130#53

Non-authoritative answer:

Name: yandex.ru

Address: 5.255.255.70

Name: yandex.ru

Address: 5.255.255.80

Name: yandex.ru

Address: 77.88.55.66

Name: yandex.ru

Address: 77.88.55.70

Из результатов видно, что поисковая система Яндекс использует несколько IP-адресов. Такой прием часто используется для обеспечения надежности высоконагруженных DNS-серверов. При этом на запрос, отправленный пользователем, отвечает наименее загруженный сервер.

3.5.1.3 Команда: **ping**

Команда работает на основе протокола ICMP и посылает на указанный сетевой интерфейсICMP-сообщение типа “эхо-запрос” (см. рис. 3.8). Если ответ поступает, то программа сообщает пользователю IP-адрес компьютера и время двойного обращения пакета (от момента отправки сообщения до момента получения ответа), иначе выводится сообщение о том, что компьютер не отвечает. Командаиспользуется для проверки работоспособности отдельных узлов сети и проверки правильности конфигурации сети, так как в выполнении этой команды участвуют система маршрутизации, DNS и сетевые шлюзы.

Обязательным аргументом команды **ping** является сетевой или доменный адрес узла, ответ от которого необходимо получить. С помощью дополнительных ключей команды можно задать число отправляемых запросов, время жизни пакета и другие параметры запроса. Ниже приведены примеры работы команды для ОС Windows и ОС Linux

**C:\>pingyandex.ru**

Обмен пакетами с yandex.ru [77.88.55.66] по 32 байт:

Ответ от 77.88.55.66: число байт=32 время=6мс TTL=59Ответ от 77.88.55.66: число байт=32 время=2мс TTL=59Ответ от 77.88.55.66: число байт=32 время=4мс TTL=59Ответ от 77.88.55.66: число байт=32 время=3мс TTL=59Статистика Ping для 77.88.55.66:Пакетов: отправлено = 4, получено = 4, потеряно = 0 (0% потерь),Приблизительное время приема-передачи в мс:Минимальное = 2мсек, Максимальное = 6 мсек, Среднее = 3 мсек

**[kvg@students ~]$ping -c 4 yandex.ru**

PING yandex.ru (5.255.255.70) 56(84) bytes of data.

64 bytes from yandex.ru (5.255.255.70): icmp\_seq=1 ttl=59 time=0.878 ms

64 bytes from yandex.ru (5.255.255.70): icmp\_seq=2 ttl=59 time=0.954 ms

64 bytes from yandex.ru (5.255.255.70): icmp\_seq=3 ttl=59 time=1.25 ms

64 bytes from yandex.ru (5.255.255.70): icmp\_seq=4 ttl=59 time=1.91 ms

--- yandex.ru ping statistics ---

4 packets transmitted, 4 received, 0% packet loss, time 3004ms

rtt min/avg/max/mdev = 0.878/1.249/1.915/0.411 ms

Обратите внимание: в Linuxдля указания количества отправленных пакетов используется ключ**-с**

3.5.1.4 Команда: **tracert**

Утилита **tracert**(в Linux–**traceroute**)работает на основе протокола ICMP и предназначена для определения маршрута следования IP-пакетов. Она отправляетэхо-запрос указанному узлу сети ивыводит сведения о всех промежуточных маршрутизаторах, через которые прошел этот запрос на пути к целевому узлу. В случае проблем при доставке данных до какого-либо узла программа позволяет определить, на каком именно участке сети возникли неполадки.

Для определения промежуточных маршрутизаторов **tracert** отправляет серию пакетов целевому узлу, при этом каждый раз увеличивая на единицузначение поля TTL, указывающее максимальное количество маршрутизаторов, которое может быть пройдено пакетом. Первый пакет отправляется при TTL=1 и поэтому первый же маршрутизатор возвращает обратно сообщение ICMPтипа 11 (истечение времени), указывающее на невозможность доставки данных. Программа **tracert**фиксирует адрес маршрутизатора иинтервал времени между отправкой пакета и получением ответа. Затем **tracert**повторяет отправку пакета при TTL=2, что позволяет первому маршрутизатору пропустить пакет дальше.Процесс повторяется до тех пор, пока пакет не достигнет целевого узла ипри получении ответа от этого узла трассировка считается завершённой.

На конечном узле IP-дейтаграмма не отбрасывается и передается программному обеспечению протокола ICMP, генерирующему сообщение типа 0 (эхо-ответ).

В Linux-системах программа **traceroute**работает иначе. Она отправляет не ICMP-сообщения типа «эхо-запрос», а UDPфрагменты с таким номером UDP-порта адресата, что он заведомо не используется на адресуемом узле (более 30000). UDPфрагменты инкапсулируются в IP- пакет, первый из которых имеет значение TTL=1, второй– равное 2 и т.д. Каждый из промежуточных маршрутизаторов выполняет декремент TTL и при значении, равным нулю, направляет отправителю ICMP-сообщение типа 11 (истечение времени). Конечный хост передает UDPфрагмент транспортному уровню, который возвращает ICMP-сообщение об ошибке «порт недоступен». Таким образом, чтобы узнать о завершении работы, программе **traceroute**достаточно обнаружить, что поступило ICMP-сообщение типа 3 с кодом ошибки, равным 3.

Утилита **traceroute**может работать в режиме передачи эхо-запросов ICMP. Для этого ее надо запустить с ключом **–I.**Ниже приведены результаты трассировки к сайту *ngs.ru* в Windows и Linux.

**C:\>tracertngs.ru**

Трассировка маршрута к ngs.ru [195.93.187.9] с максимальным числом прыжков 30: 1 <1 мс <1 мс <1 мс 192.168.1.1 2 <1 мс <1 мс <1 мс 192.168.0.1 3 4 ms 4 ms 4 ms NVSK-BRAS3.sib.ip.rostelecom.ru [213.228.116.9] 4 4 ms 4 ms 4 msae9.NVSK-RGR5.sib.ip.rostelecom.ru [213.228.109.18] 5 2 ms 4 ms 4 msae3.NVSK-AR1.sib.ip.rostelecom.ru [213.228.109.51] 6 1 ms 4 ms 4 ms 1889.egw2.core.ngs.ru [195.93.186.2] 7 4 ms 4 ms 4 ms 195.93.186.5 8 4 ms 4 ms 4 mswebcluster.ngs.ru [195.93.187.9] Трассировка завершена.

**[kvg@students ~]$traceroute ngs.ru**

traceroute to ngs.ru (195.93.187.9), 30 hops max, 60 byte packets

1 gw-130-208v.ami.nstu.ru (217.71.130.251) 3.763 ms 4.005 ms 4.315 ms

2 gate.ami.nstu.ru (217.71.130.129) 0.379 ms 0.424 ms 0.447 ms

3 ix-i.nstu.ru (217.71.131.1) 1.086 ms 1.328 ms 1.294 ms

4 c6k-720.nstu.ru (217.71.128.67) 1.239 ms 1.161 ms 1.131 ms

5 nsk-ix.nstu.ru (217.71.128.193) 1.293 ms 1.263 ms 1.333 ms

6 egw4.core.ngs.ru (193.232.87.32) 1.108 ms 1.056 ms 1.029 ms

7 195.93.186.5 (195.93.186.5) 1.654 ms 1.729 ms 1.587 ms

8 webcluster.ngs.ru (195.93.187.9) 1.155 ms 1.129 ms 1.099 ms

3.5.1.5 Команда: **netstat**

Утилита**netstat**предназначена длявывода статистики по протоколам и подключениям TCP/IP. При вызове с ключом **–а** программа выводит информацию о состоянии всех активных сокетов с указанием локального и внешнего адресов (ESTABLISHED – соединение установлено, TIME\_WAIT - сокет закрыт, но ожидает пакеты, еще находящиеся в сети для обработки, LISTENING – сокет ожидает входящих соединений и т.д.).

Ключ **–r** указывает на необходимость вывода таблицы маршрутизации, а ключ **–s** – необходимость вывода статистики по протоколам. В Linux команда **netstat** с ключом –i выводит информацию о сетевых интерфейсах.

Ниже приведен пример вывода статистики по протоколу IPv4 в Windows и Linux.

**C:\>netstat -s**

Статистика IPv4

Получено пакетов = 18907

Получено ошибок в заголовках = 0

Получено ошибок в адресах = 58

Направлено датаграмм = 0

Получено неизвестных протоколов = 0

Отброшено полученных пакетов = 138

Доставлено полученных пакетов = 18769

Запросов на вывод = 19183

Отброшено маршрутов = 0

Отброшено выходных пакетов = 0

Выходных пакетов без маршрута = 0

Требуется сборка = 0

Успешная сборка = 0

Сбоев при сборке = 0

Успешно фрагментировано датаграмм = 0

Сбоев при фрагментации датаграмм = 0

Создано фрагментов = 0

Статистика TCP для IPv4

Активных открыто = 781

Пассивных открыто = 87

Сбоев при подключении = 3

Сброшено подключений = 121

Текущих подключений = 91

Получено сегментов = 17124

Отправлено сегментов = 12765

Повторно отправлено сегментов = 4061

Статистика UDP для IPv4

Получено датаграмм = 1529

Отсутствие портов = 116

Ошибки при получении = 0

Отправлено датаграмм = 2336

**[kvg@students ~]$netstat -s**

Ip:

76013304 total packets received

0 forwarded

0 incoming packets discarded

67453711 incoming packets delivered

39648867 requests sent out

55 dropped because of missing route

**[kvg@students ~]$netstat -i**

Kernel Interface table

Iface MTU RX-OK RX-ERR RX-DRP RX-OVR TX-OK TX-ERR TX-DRP TX-OVR Flg

eth0 1500 101285588 0 0 0 32106551 0 0 0 BMRU

lo 65536 11707980 0 0 0 11707980 0 0 0 LRU

virbr0 1500 0 0 0 0 0 0 0 0 BMU

### 3.5.2 Интерактивные средства диагностики IP-сетей

В настоящее время имеется достаточно много различных интерактивных сетевых сервисов, позволяющих решить проблемы пользователей, возникающие при работе с сетью. Такие сервисы условно можно разделить на две группы: предназначенные для сетевых программистов и предоставляющие услуги (хостинг или облачное хранение данных).

К первой группе относятся, например, сервисы **network-tools.com, speed-tester.info, 24whois.ru, seogadget.ru, ip-ping.ru**. Они содержат набор аналитических инструментов для веб-мастеров и позволяют проверить пинг, выполнить трассировку, определить IP-адрес по заданному доменному имени, получить информацию по указанному IP или доменному адресу, включая его местонахождение, а также выполнить другие аналогичные задачи.

К этой же группе можно отнести сервисы, выполняющие трассировку от указанного пункта в любой точке мира к компьютеру пользователя и отображающие эту трассу на географической карте. Такими сервисами являются, например,

**ping-admin.ru, monitis.com/traceroute, yougetsignal.com/tools/visual-tracert**идр.

Для сервисов второй группы средства диагностики сетей являются вспомогательными и в основном содержат функцию *whois*, с помощью которой можно получить информацию о владельце сайта, дате регистрации и дате окончания регистрации доменного имени. Примером такого сервиса является **whois.com**.

## 3.6 Практическое задание

Задание выполняется в два этапа:на первом этапе необходимо изучить основные способы диагностики IP-сетей, на втором – спроектировать и реализовать программу, выполняющую основные функции утилит **ping** и **tracert**.

Задания первого этапа выполняются в средах ОС Windows (терминальный компьютер) и Linux (удаленный сервер).

Этап 1

1. Подключитесь к серверу и с помощью команды **uname**получитеполную информацию об установленной ОС и аппаратной платформе.

2. Получите статистику по сетевым интерфейсам компьютера и сервера, проведите анализ результатов.

3. Просмотрите содержимое DNS-кэша компьютера и сервера, поясните характеристики записей и очистите кэш.

4. Просмотритесодержимое ARP-таблиц компьютера и сервера, пояснитехарактеристики записей, выполните добавление и удаление статических записей.

5. Просмотритесодержимое таблиц маршрутизации компьютера и сервера, поясните характеристики записей.

6. В командном режиме на компьютере и сервере определитеIP-адреса поисковых систем *google.com, yandex.ru, rambler.ru, ya.ru*и поясните результаты.

7. В командном режиме на компьютере и на сервере определитеIP-адреса узлов*mit.edu, vk.com, stanford.edu, ox.ac.uk, telegram.com*, выполнить для них пингование и трассировку.

8. С помощью интерактивных сетевых сервисов (например, **ping-admin.ru, whois.com**) выполните трассировку, определите местонахождение и владельца узлов сети, указанных в п.7. Выполните анализ полученных результатов.

Этап 2

Реализовать Windows- илиLinux-приложение, которое будет выполнять основные функции утилит **ping**и **tracert**. Результат работы приложения сравните с результатами, полученными при выполнении п.7 первого этапа.

## 3.7 Контрольные вопросы

1. Как определить доступность рабочей станции в сети Интернет?
2. Как определить количество маршрутизаторов на пути от вашего компьютера до требуемого вам узла сети ?
3. На каком уровне модели OSIработают программы **tracert**и**traceroute**?
4. Назовите основные типыICMP – сообщений.
5. Назначение поля «Код» в ICMP – сообщении.
6. Основные возможности стандартной утилиты **ping**.
7. Алгоритм работы утилиты **traceroute**.
8. Структура ICMPпакета.
9. Схема инкапсуляции ICMP-пакета в кадрEthernet.

4. ТРАНСПОРТНЫЙ УРОВЕНЬ

*Прочитав эту главу, вы узнаете:*

* *основы работы транспортных протоколов TCP и UDP;*
* *принципы удаленного взаимодействия между процессами;*
* *структуру TCP-сегмента и UDP-дейтаграммы;*
* *определение следующих терминов:сокет, порт, сегмент, мультиплексирование, демультиплексирование, тройное рукопожатие, скользящее окно.*

Успешный обмен сообщениями между процессами, запущенными на двух удаленных компьютерах, возможен только тогда, когда процессы могут идентифицировать друг друга. Для этого необходимо знать IP-адрес компьютера, на котором запущен процесс, и номер порта, связанного с процессом. Адрес используется на сетевом уровне, чтобы доставить пакет на указанный узел, а номер порта используется на транспортном уровне для доставки данных указанному процессу.

Транспортный уровень обеспечивает передачу данных между удаленными процессами, каждый из которых взаимодействует с транспортным уровнем с помощью специального интерфейса программирования приложений (API) – сокета. Сокет представляет собой библиотеку функций, обеспечивающих заданные параметры сеанса обмена данными и управляющих передачей и приемом данных.С логической точки зрения сокет является абстрактным объектом, представляющим точку подключения приложения к сети.

Параметры сокета задаются разработчиком сетевого приложения (тип транспортного протокола, IP-адрес, номер порта, максимальный размер буфера, максимальный размер сегмента и т.д.). Напомним, что единицей данных транспортного уровня для протокола UDP является дейтаграмма, а для протокола TCP – сегмент.

Протоколы TCP и UDP для каждого приложения создают две очереди – для приема и для отправки данных. С точки зрения программиста совокупность этих очередей является портом. Протоколы TCP и UDP используют собственные пространства номеров для портов, т.е. TCP-порт 53 и UDP-порт 53 – это разные порты, которые могут использоваться протоколом DNS.

Модули TCP и UDP выполняют функции мультиплексоров/демультиплексоров между прикладными процессами и IP-модулем.**Мультиплексирование** – это объединение нескольких отдельных потоков данных в общий агрегированный поток, который можно передавать по одному каналу связи, **демультиплексирование** – обратная операция.Алгоритмы обработки входного потока байтов и структура дейтаграммы (сегмента) зависят от используемого транспортного протокола (UDP или TCP) и функции, выполняемой узлом в данный момент –отправитель или получатель данных.

**На стороне отправителя** транспортный уровень получает от прикладного уровня сообщение в виде потока байтов. Далее выполняются следующие действия:

1. на уровне сокета вызывается служба UDP или TCP;

2. данные копируются из рабочей области приложения в буфер передачи сокета:

- TCP копирует данные в свой буфер сокета;

- UDP копирует данные в свой буфер и вычисляет контрольную сумму;

3. если размер данных превышает размер максимального блока передачи (MTU) локальной сети, то:

- TCP разбивает данные на фрагменты, размер которых не превышает размера MTU;

- для UDP фрагментация данных выполняется на следующем уровне - уровне IP.

4. при необходимости IP разбивает данные на блоки таким образом, чтобы размер исходящих пакетов не превышал ограничения MTU.

5. пакеты помещаются в кадры канального уровня, которые передаются сетевым адаптером в сеть;если очередь вывода сетевого адаптера переполнена, то пакет отбрасывается.

**На стороне получателя** выполняются следующие действия:

1. поступающий поток битов помещается в очередь сетевого адаптера, который средствами канального уровня формирует кадры и передает их сетевому уровню;

2. если сетевой уровень определяет, что отправитель разбил данные на фрагменты, то он собирает фрагменты в исходный блок данных и передает его транспортному уровню (TCP или UDP):

- TCP при необходимости объединяет сегменты данных и помещает результат в буфер приема сокета;

- UDP передает полученные данные в буфер приема сокета;если буфер приема сокета переполнен, то пакет отбрасывается.

3. когда приложение отправляет запрос на чтение, необходимые данные копируются из буфера приема сокета в буфер приложения.

## 4.1 Протокол UDP

Данные с прикладного уровня поступают через порты каждого процесса на транспортный уровень в виде сообщений.К каждому сообщению UDP добавляет заголовок, формируя UDP-дейтаграммы,которые передаются на сетевой уровень в режиме мультиплексирования (рис.4.1).

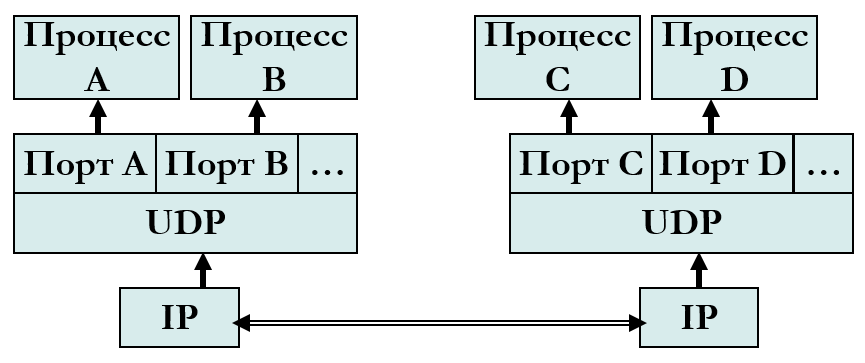
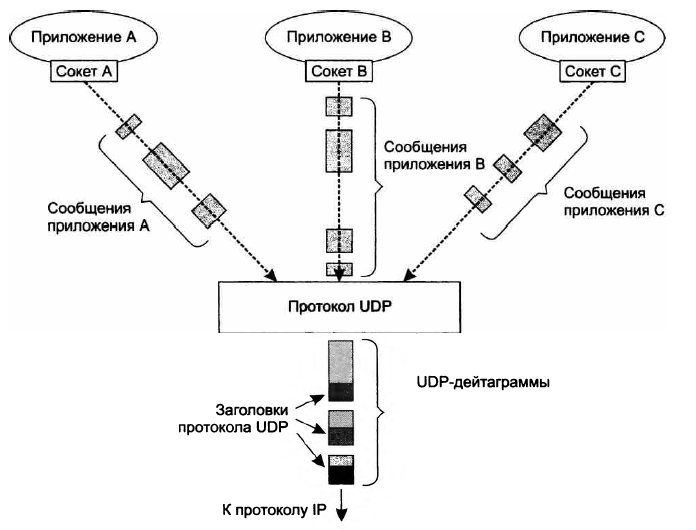


Рис. 4.1

Протокол UDP сохраняет границы сообщений, заданные прикладным процессом. Он никогда не объединяет несколько сообщений в одно и не делит одно сообщение на части.

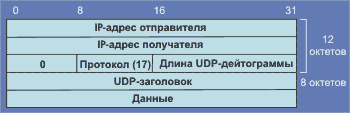
На рис. 4.2а показан общий формат UDP-дейтаграммы, заголовок которой содержит 4 поля размером 2 байта каждое:

- номер UDP-порта отправителя;

- номер UDP-порта получателя;

- контрольная сумма;

- длина дейтаграммы с учетом заголовка.



а) б)

Рис. 4.2

Контрольная сумма содержит число, полученное в результате контрольного суммирования UDP-заголовка и поля данные.Протокол UDP проводит только диагностику ошибок, не исправляя их.Если контрольная сумма показывает ошибку, то дейтаграмма отбрасывается.

Если протокол UDPработает в паре с протоколом IP, то формат UDP-дейтаграммы изменяется за счет добавления псевдозаголовка, используемого для контроля отсутствия ошибок в IP-адресах (рис. 4.2б).Псевдозаголовок формируется из модуля IP и содержит IP-адреса отправителя и получателя, код протокола UDP (17) и длинуUDP-дейтограммы. Контрольная сумма в этом случае вычисляется с учетом UDP-заголовка (8 байт) и псевдозаголовка (12 байт). Длина псевдозаголовка в общей длине дейтаграммы не учитывается.

Максимальный размер UDP-дейтаграммы теоретически составляет 65535 байтов, реальный размер передаваемых данных не может превышать 65507 байтов, так как8 байт занимает UDP-заголовок и 20 байтов – заголовок IPv4.

Протокол UDPиспользуется в качестве транспорта для прикладных протоколовDNS, DHCP, RIPи других.

## 4.2 Протокол TCP

Основным достоинством протокола TCP является возможность гарантированной доставки всех пакетов получателю при использовании ненадежного дейтаграммного протокола IP.Гарантии реализуютсяпредварительным установлением логического соединения между отправителем и получателем, а также применением многоступенчатого контроля корректности полученных данных. При этом проверяется очередность поступления пакетов и их целостность. Если пакет потерялся или в нем при прохождении по сети возникла ошибка, то получатель запрашивает повторную передачу у отправителя.

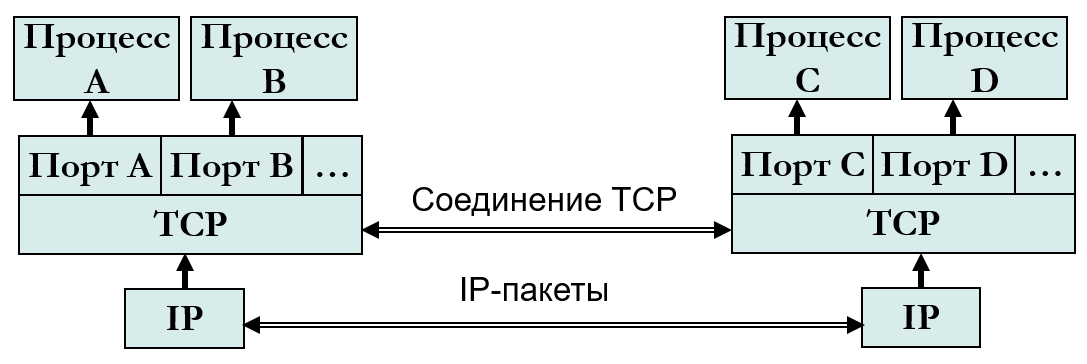




Рис. 4.3

На стороне отправителя на вход модуля TCP подаются потоки байтов от приложений прикладного уровня (рис. 4.3). TCP разбивает потоки байтов каждого соединения на сегменты, которые в режиме мультиплексирования передаются на сетевой уровень.

TCP устанавливает между двумя процессами логическое соединение, которое идентифицируется парой адресов сокетов IP1,n1 и IP2,n2., где IP и n – это IP-адрес и номер порта, связанные с сокетом.На этапе установления соединения стороны договариваются о параметрах соединения (максимальный размер сегмента, максимальный объем данных, который можно передать без подтверждения приема, начальный номер байта, с которого начинается отсчет потока данных).

### 4.2.1 Установление и разрыв соединения

Процедуры установления и разрыва TCP-соединения показаны на рис. 4.4.

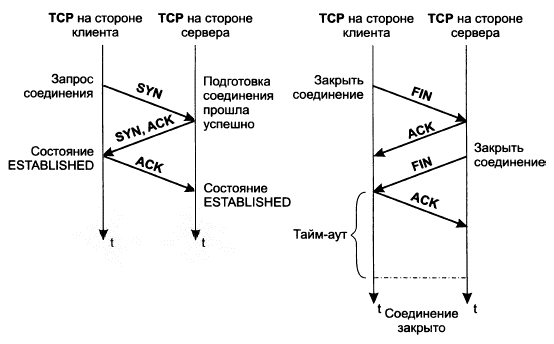


Рис. 4.4

Установление соединения часто называется «тройное рукопожатие», инициализируется клиентом и начинается с запроса на синхронизацию счетчиков переданных байтов SYN, в котором указывается порядковый номерначального байта в своем потоке данных. Получатель (сервер) в ответ передаёт сообщение SYN, которое содержит подтверждение получения предыдущего сообщения ACK, порядковый номер байта, который он ожидает и номер своего начального байта.Клиент переходит в состояние установленного логического соединения (ESTABLISHED) и в ответ направляет подтверждение ACK с подтверждением принятых начальных номеров байтов каждой стороны, увеличивая их на 1. После этого в состояние ESTABLISHEDпереводится сервер, соединение считается установленным и можно передавать данные.

Протокол TCP является дуплексным, то есть каждыйучастник соединения может передавать и принимать данные одновременно, поэтому инициировать разрыв соединения может любая сторона. Другая сторона при этом может по прежнему передавать данные.

На рис. 4.4 разрыв соединения начинает клиент, передавая сообщение FIN, которое говорит об окончании передачи данных клиентом. В ответ сервер подтверждает односторонний разрыв соединения сообщением ACK, но может продолжить передачу данных клиенту, если она не завершена. По окончании передачи сервер также посылает клиенту сообщение FINо готовности разорвать соединение со своей стороны, на которое клиент отвечает подтверждением ACK.

### 4.2.2 Обмен данными

Для обеспечения надежности передачи данных в сетях применяются два мето-да, основанных на использовании специальных сообщений(квитанций), подтверждающих прием: остановка с ожиданием и скользящее окно. В первом случае источник должен ожидать получения квитанции после отправки каждого пакета данных и только после этого отправлять следующий пакет. Это приводит к существенному снижению скорости обмена данными.

Применение скользящего окна дает возможность источнику передавать несколько пакетов данных без ожидания подтверждающих квитанций для каждого пакета. Объем данных, которые могут быть переданы без подтверждения, называется размером окна. Окно включает в себя определенную часть входного потока данных и задает размер сегмента, формируемого протоколом TCP. Один сегмент TCPможет переносить пользовательские данные и квитанцию (в поле заголовка).Размер окна на каждой стороне TCP-соединения формируется в ходе установления соединения и может динамически изменяться в процессе обмена данными.

На рис. 4.5 приведен формат TCP-сегмента, имеющего следующие поля:

* код позиции в сообщении – идентификатор сегмента (порядковый номер первого байта);
* номер октета, который должен прийти следующим – номер начального байта следующего ожидаемого сегмента;
* hlen –длина заголовка сегмента, которая измеряется в 32-разрядных словах;
* размер окна - сообщает, сколько байтов готов принять получатель. Определяет объем данных, которые могут быть посланы без получения подтверждения. Ширина окна может изменяться во время обмена данными;
* флаги – набор битов, определяющих характер передаваемых данных: SYN – синхронизация номеров последовательности передаваемых байтов; ACK – квитанция на принятый сегмент; FIN - сообщение о разрыве соединения, RST – запрос на восстановление соединения и т.д.
* контрольная сумма – контрольная сумма, вычисляемая по содержимому всего сегмента и псевдозаголовка, содержащего IP-адреса отправителя и получателя, а также размер TCP-сегмента; псевдозаголовок при передаче не используется.
* указатель важной информации -  номер первого байта срочных данных (в поле данных сегмента).

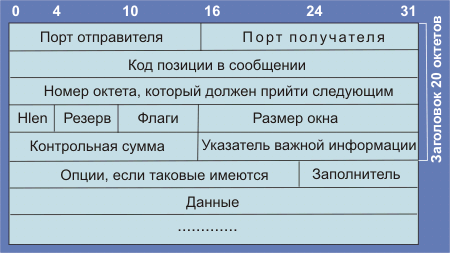
******

Рис. 4.5

Для контроля отсутствия ошибок в IP-адресах по аналогии с протоколом UDPв начало сегмента добавляется псевдозаголовок размером 12 байтов, в котором содержатся IP-адреса отправителя и получателя, код протокола TCP (6) и длина сегмента без учета псвдозаголовка (см. рис. 4.2 б).

На рис. 4.6 приведен пример передачи сообщения с применением окна размером 1400 байтов. Здесь идентификатором каждого сегмента является номер его начального байта, то есть первый сегмент имеет идентификатор 0, второй – 1400 и т.д. Сначала формируется первый сегмент, номер его начального байта записывается в заголовок и сегмент отправляется получателю. Получатель после контроля правильности полученных данных формирует квитанцию, в которую вкладывает номер следующего ожидаемого байта –1400. Отсутствие квитанции означает прием искаженного сегмента, потерю сегмента или потерю квитанции.

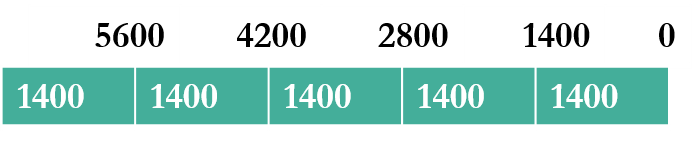


Рис. 4.6

На каждой стороне ТСР-соединения есть 3 буфера, которые выделяются при создании сокетов (рис. 4.7):

- буфер приема для хранения принятых сегментов;

- буфер отправления для хранения сегментов, которые надо отправить;

- буфер копий для хранения копийотправленныхсегментов, квитанции по которым еще не получены.

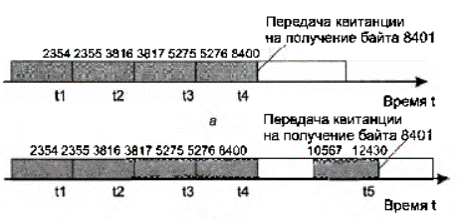
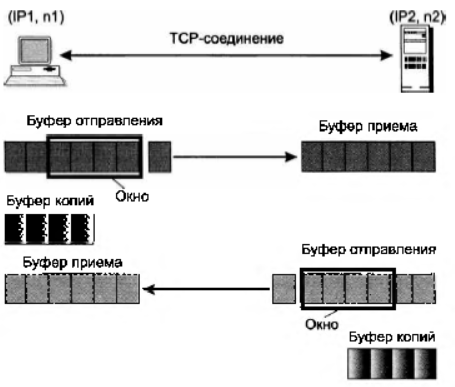


Рис. 4.7

Квитанции должны передаваться последовательно. Например, если возникает пропуск сегмента 8401-10566 байтов (см. рис. 4.6), то после приема сегмента 10567-12430 байт снова передается квитанция на 8401 байт, то есть приемная сторона просит повторить повторную передачу сегментов, начиная с байта 8401. Такой способ называется накопительным квитированием.

Протокол TCPиспользуется в качестве транспорта для прикладных протоколовDNS, HTTP, FTP, SSHи других.

## 4.3 Контрольные вопросы

1. Транспортный уровень: назначение, система идентификации отправителей и получателей данных.
2. Протоколы транспортного уровня: назначение, достоинства и недостатки.
3. Поясните понятия мультиплексирование и демультиплексирование
4. Назовите порядок обработки данных протоколом UDPна стороне отправителя.
5. Назовите порядок обработки данных протоколом TCPна стороне отправителя.
6. Назовите порядок обработки данных протоколом UDP на стороне получателя.
7. Транспортный уровень: назначение, система идентификации отправителей и получателей данных.
8. Протоколы транспортного уровня: назначение, достоинства и недостатки.
9. Поясните понятия мультиплексирование и демультиплексирование
10. Назовите порядок обработки данных протоколом UDPна стороне отправителя.
11. Назовите порядок обработки данных протоколом TCPна стороне отправителя.
12. Назовите порядок обработки данных протоколом UDPна стороне получателя.
13. Назовите порядок обработки данных протоколом TCPна стороне получателя.
14. UDP-дейтаграмма:формат, назначение полей.
15. TCP-сегмент:формат, назначение полей.
16. Псевдозаголовок транспортного уровня: назначения и структура.
17. Порядок установления логического соединения и его разрыва.
18. Скользящее окно: понятие и принцип использования.
19. Может ли протокол TCPиспользоваться для широковещательной рассылки?

5. ОСНОВЫ РАЗРАБОТКИ СЕТЕВЫХ ПРИЛОЖЕНИЙ

*Прочитав эту главу, вы узнаете:*

* *схему взаимодействия клиента и сервера;*
* *основные функции для работы с сокетами;*
* *особенности разработки многопоточного сервера;*
* *определение следующих терминов: клиент, сервер, коммуникационный домен, многопоточность, семафор,мутекс*

## 5.1 Общие сведения

Сетевое приложение, как правило, состоит из двух компонентов – серверного и клиентского.**Сервер** представляет собой программу, которая реализует основные функции сетевого приложения и после загрузки пассивно ожидает поступления запросов на установление соединения от удаленных клиентов. Обычно такая программа разрабатывается с учетом возможности одновременного обслуживания нескольких удаленных клиентов и размещается на мощном компьютере, предоставленном в общее пользование.

**Клиент** – это программа, которая запускается на локальном компьютере, обращается к удаленному серверу с определенными запросами и выполняет обработку результатов этих запросов. Клиент может обращаться по мере необходимости к несколькимсерверам, но в любой момент времени активно контактирует только с одним из них.

Общая схема взаимодействия клиента и сервера приведена на рис. 5.1.

Ожидание запроса на соединение от клиента

Установление соединения с клиентом

Прием данных от клиента

Обработка полученных данных

Пересылка данных клиенту

Разрыв соединения с клиентом

Передача серверу запроса на соединение

Пересылка данных серверу

Прием данных от сервера

Отправка сообщения об окончании приема данных

**Сервер**

**Клиент**

Рис. 5.1

Для организации взаимодействия сетевые приложения используют специальные объекты – сокеты, представляющие собой конечные точки логического соединенияклиента и сервера. Сокеты работают на транспортном уровне модели OSI и позволяют скрыть от программиста все проблемы, решаемые на сетевом, канальном и физическом уровнях. Достаточно только создать сокет, правильно установить его параметры и корректно применять все методы, предоставляемые специальным API-интерфейсом (например, WinSock32).

Основными параметрами сокета являются IP-адрес и номер порта, причем для стандартных протоколов Internetзарезервированы номера портов в диапазоне (0 – 1023), а для разработчиков доступны номера, начиная с 1024. Комбинация IP адреса и номера порта однозначно определяет отдельный сетевой процесс во всей глобальной сети Internet.

Для создания соединения TCP/IP необходимы два сокета – один на локальной машине, а другой на удаленной. Таким образом, каждое сетевое соединение идентифицируется двумя парами чисел – IP-адресом и портом на локальной машине, IP-адресом и портом на удаленной машине.

Работа с сокетами может проводиться на основе библиотек высокого или низкого уровней. Применение библиотек высокого уровня существенно упрощает работу программиста, но снижает уровень понимания всех этапов организации сетевого взаимодействия процессов. Поэтому в дальнейшем описании мы будем использовать API-интерфейс низкого уровня.

## 5.2 Функциидля работы с сокетами

### 5.2.1 Функцияsocket

Функцияsocket создает объект типа сокет и возвращает его дескриптор:

s = socket(af, type, [protocol]);

Параметр *af*задает имя **коммуникационного домена**, используемого сокетоми может принимать два значения: AF\_INET(или PF\_INET) для домена Internetи AF\_UNIX(или AF\_LOCAL) для домена UNIX. Коммуникационный домен определяет систему адресации соединений.

Домен Internetиспользуется для организации обмена данными между удаленными компьютерами, а домен UNIX – для межпроцессного обмена данными на локальном компьютере. Для домена Internetадрес записывается в виде IP-адреса и номера порта, а домен UNIXиспользует файловую систему как адресное пространство имендля организации межпроцессного обмена данными на локальном компьютере. При этом сокет UNIX представляется процессам в виде дескриптора файла с заданным именем, что позволяет двум различным процессам открывать один и тот же сокет для взаимодействия между собой, но реально обмен данными проводится с использованием буферов ядра ОС, а не внешней памяти.Далее мы будем рассматривать только сокеты, работающие в домене Internet.

Параметр *type* указывает способ передачи данных, который будет использоваться сокетом. Наиболее распространенными типами являются потоковая передача с установлением логического соединения (SOCK\_STREAM) и дейтаграммная передача без установления логического соединения (SOCK\_DGRAM).

Необязательный параметр *protocol* определяет тип транспортного протокола и для Internet-доменаможет принимать значения IPPROTO\_TCP илиIPPROTO\_UDP.

### 5.2.2 Функция bind

Сразу после создания сокет ассоциируется с некоторым семейством адресов, но не с конкретным адресом.Для сокетов, работающих в Internet-домене, адрес представляется парой (*host, port*), где *host* указывает IP-адрес хоста, а *port* – номер порта.Для UNIX-домена адрес указывается в виде константы SOCKET\_FILE.

Для привязки сокета к конкретному адресу применяется функция *bind*:

retVal = bind(s, (LPSOCKADDR)&sin, sizeof(sin));

Здесь s – дескрипторсокета, (LPSOCKADDR)&sin – адресструктуры sin, в которой задается конкретный адрес привязки, sizeof(sin) – размер структуры sin.Структура имеет тип SOCKADDR\_IN и описывается следующим образом:

SOCKADDR\_IN sin;

sin.sin\_family = AF\_INET;

sin.sin\_port = htons(port);

sin.sin\_addr.s\_addr = host;

Имя хоста может задаваться в виде доменного имени (например, ‘localhost’) или в виде IP-адреса.

На этом этапе сокет связывается с сетевым интерфейсом компьютера, что особенно важно для сервера, так как при наличии нескольких сетевых интерфейсов может возникнуть ситуация неопределенности. Например, если в компьютере установлены две сетевые карты, одна из которых подключена к локальной сети, а другая к сети Интернет, и сокет будет связан с локальным адресом компьютера, то сервер сможет принимать запросы только из локальной сети. Если связать сокет с внешним (белым) адресом, то сервер сможет принимать запросы только из внешней сети Интернет.

Для того, чтобы сервер мог принимать запросы как из локальной сети, так и из внешней сети, а также из внутреннего стека TCP/IP через внутреннюю петлю localhost, необходимо связать сокет со специальным IP-адресом 0.0.0.0, который можно задать явно или с помощью константы INADDR\_ANY. При этом сокет зарегистрируется сразу на всех сетевых интерфейсах компьютера.

### 5.2.3 Функция listen

После привязки сокета к адресу сервер переводится в режим прослушивания порта и ожидания запросов на установление соединения от клиентов. Для этого сервер вызывает функцию*listen*, принимающуюдва параметра – дескриптор сокета и максимальный размер очереди запросов на соединение:

retVal = listen(s, queuesize);

### 5.2.4 Функцияaccept

Обработка очереди запросов проводится функцией *accept*:

s\_client = accept(s, (struct sockaddr\*)&from, &fromlen)

Здесь*s* – дескрипторслушающегосокета, *(structsockaddr\*)&from* – структура, содержащая адресную информацию о клиенте, *&fromlen* – размер структуры *from*. Функция создает новый сокет, через который будет проводиться обмен данными с клиентом и возвращает его дескриптор*s\_client*. При этом слушающий сокет сервера по прежнему работаетв режиме прослушивания порта для получения новых запросов на соединение от клиентов.

### 5.2.5 Функция connect

В клиентских программах для установления соединения с сервером применяется функция*connect*, которой передаются следующие параметры: *clientSock* – дескриптор сокета клиента, *(LPSOCKADDR)&serverInfo* – структура, аналогичная описанной в разделе 5.2.2 и содержащая адрес сервера, *sizeof(serverInfo)* – размер этой структуры:

retVal=connect(clientSock,(LPSOCKADDR)&serverInfo, sizeof(serverInfo))

Если сокет не был связан с адресом, то *connect* автоматически вызовет системную функцию *bind*.

### 5.2.6 Функции send и sendto

Функции *send* и *sendto*используются для передачи данных. Если сокет подключен, то для передачи данных применяется функция*send*, которая имеет следующие параметры:*s* – дескрипторсокета, через который проводится передача; *pBuf* и *strlen(pBuf)* – имя и размер буфера, содержащего данные, *flags*– список констант, с помощью которых можно запрашивать специальные опции.

retVal = send(s, pBuf, strlen(pBuf), flags)

Функция*sendto* позволяет передавать сообщение с использованием неподключенного сокета. Параметры этой функции совпадают с параметрами функции *send*, к которым добавляется адрес получателя:

rv =sendto(s,pBuf,strlen(pBuf),flags,(LPSOCKADDR)&serverInfo, sizeof(serverInfo))

Обе функции возвращают количество переданных байтов.

### 5.2.7 Функции recv и recvfrom

Функции recv и recvfrom используются для приема данных. Для приемаданных из подключенного сокета используется функция *recv*:

retVal = recv(s, pBuf, strlen(pBuf),flags)

Здесь *s* – дескрипторсокета, из которогочитаются данные; *pBuf* и *strlen(pBuf)* – имя и размер буфера для приема данных, *flags*– список констант, с помощью которых можно запрашивать специальные опции.

Функция *recvfrom* дает возможность прочитать данные из сокета с указанным адресом

rv = recvfrom (s,pBuf,strlen(pBuf),flags,(LPSOCKADDR)&serverInfo, sizeof(serverInfo))

Необязательный параметр *flags* аналогичен этому же параметру методов send и sendto. Обе функции возвращают количество принятых байтов.

### 5.2.8 Функцияclose

Функцияclose сообщает ОС, что программа завершила использование сокета. Вызов имеет форму:

closeSocket(s)

Схема взаимодействия сервера с клиентом через интерфейс сокетов приведена на рисунке 5.2.

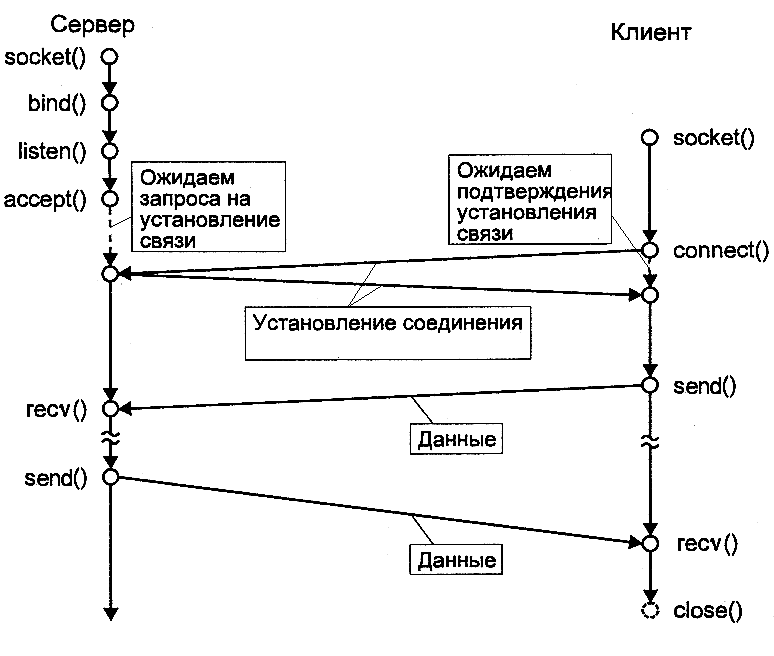


Рис. 5.2

## 5.3 Особенности разработки многопоточного сервера

### 5.3.1 Общие сведения

Выше был рассмотрен сервер, который проводит последовательную обработку поступающих от клиентов запросов. В случае занятости сервера поступающий запрос ставится в очередь, размер которой задается в функции listen.

Для сокращения времени ожидания клиентов и повышения производительности сервер должен обслуживать одновременно несколькозапросов клиентов, создавая для каждого из них отдельный сокет. При этом возможны следующие варианты обработки поступающих запросов:

* применить неблокирующие сокеты и читать в цикле данные из каждого сокета поочереди,
* для каждого клиента запускать отдельный поток выполнения, который будет заниматься обменом данными с клиентом.

С точки зрения производительности сервера более предпочтительным является второй вариант, потому рассмотрим основные сведения о многопоточном режиме выполнения программ.

Любой процесс, создаваемый операционной системой (ОС), представляет собой набор команд программы, запускаемой пользователем, и метаданных, представленных набором выделенных системных ресурсов и общих сведений о запускаемой программе. В качестве ресурсов выступают адресное пространство, глобальные переменные и открытые файлы. Общими сведениями являются идентификаторы текущего, родительского и дочерних процессов, а также идентификаторы владельца процесса и группы, членом которой является владелец.

Поток выполнения – это поименованный набор команд программы, который является минимальной единицей управления для ядра ОС и создается в рамках процесса. Многозадачный режим обработки программ ОС реализует путем переключения процессора между готовыми к выполнению потоками.

В однопоточном приложении есть только один поток выполнения команд и при запуске программы все команды исполняются последовательно с учетом возможных ветвлений и циклов. В этом случае поток выполнения непрерывен в течение всего времени выполнения программы и завершается только по завершении программы.

В многопоточных приложениях возможна организация нескольких потоков выполнения, которые будут исполняться процессором параллельно. Программной единицей для организации отдельного потока является функция, т.е. основной поток будет как обычно представлен функцией **main()**, а в качестве других потоков могут выступать другие функции, описанные в программе.

Любой поток выполнения, работающий в многопоточном приложении, имеет доступ ко всем ресурсам, выделенным процессу. Например, несколько потоков могут пытаться одновременно изменить содержимое одного участка памяти. Поэтому при разработке многопоточных приложений особое внимание необходимо уделять синхронизации потоков при работе с общими ресурсами, для чего наиболее удобным является применение семафоров.

Семафоры позволяют исключить выполнение одного и того же участка кода программы несколькими потоками одновременно. Самый простой двоичный семафор – замок (lock) или мутекс (mutex). Чтобы поток мог продолжить исполнение защищенного фрагментакода, он должен сначала получить замок и закрыть его. После этого он выполняет команды этогофрагментаи открывает замок, чтобы другой поток (возможно, уже сделавший запрос на данный замок) мог тоже исполнять этот код.

Необходимые для работы с потоками функции содержатся в модулях **thread** и **threading**.

### 5.3.2Управление потоками

Объект**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:**

**start()** запускает поток, он не имеет параметров и ничего не возвращает;

**run()** используется для вызова в отдельном потоке. Получает свое значение из параметра **target** конструктора класса,при вызове используются **args**и **kwargs**, заданные в конструкторе;

**join([time])**ожидает завершения потока. Поток, который вызывает этот метод, приостанавливается,значение **time** задает время ожидания, после которого приостановленный поток продолжает свою работу;

**getName()**возвращает имя потока;

**setName(name)**устанавливает имя потока;

**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 три основных метода:

**acquire()**запирает замок;

**release()**открывает замок;

**locked()**возвращает статус замка: 0 – открыт, 1 – закрыт.

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 представлены классом **threading.Semaphore**иявляются более общим механизмом синхронизации потоков, чем замки. С их помощью в критическую часть программы допускаются 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 выполнения. В этом модуле доступны следующие функции:

**thread.start\_new(function,args[,kwargs]) –** создание нового потока для исполнения функции *function*с кортежем аргументов *args*и словарем*kwargs*, поток завершается по возврату из функции;

**thread.exit()**– возбуждает исключение **SystemExit,** котороезавершает исполняющийся поток, если не оказалось перехваченным;

**get\_ident() ->id**– возвращает идентификатор потока, в котором исполняется эта функция;

**thread.allocate\_lock() ->lockobj**– создание замка,на основе которого создан класс **threading.Lock**, поэтому методы для работы с этими объектами одинаковы.

## 5.4 Практическое задание

### 5.4.1 Разработка однопоточного сервера

Напишите простейшее приложение, реализующее удаленное взаимодействие клиент-сервер, используя API-интерфейс низкого уровня. Клиент должен направить серверу определенный запрос, а сервер – дать ответ на этот запрос.

При запуске сервера необходимо указать номер порта протокола TCP или UDP, не забывая о том, что пользовательский диапазон номеров начинается с 1024. Сервер должен создать прослушивающий сокет, связать его с IP-адресом и номером порта и перейти в режим ожидания запроса от клиента на соединение.

Запуск клиента происходит с указанием IP-адреса и номера порта сервера. Клиент устанавливает связь с сервером и посылает ему набор данных, введенных пользователем. Сервер получает набор данных, выполняет указанные в варианте задания действия, возвращает клиенту результат и снова переходит в состояние ожидания запроса на соединение. Клиент, получив ответ с сервера, выводит его на экран и прекращает свою работу.

Тестирование совместной работы сервера и клиента необходимо провести в режиме удаленного доступа. Если отсутствует возможность использовать для тестирования два компьютера с «белыми» IP-адресами, то можно создать виртуальную сеть на основе бесплатной VPN-технологии Hamachi, которая является надстройкой над сетью Интернет и может использоваться для объединения в единую защищенную сеть компьютеров с «серыми»IP-адресами. Дистрибутив Hamachi можно скачать по адресу <http://hamachi-pc.ru/>

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

Пример реализации приложения, в котором клиент передает серверу два числа, а сервер возвращает их сумму, приведен в приложении 1.

### 5.4.2 Разработка многопоточного сервера

На основе разработанного однопоточного сервера с помощью API-интерфейса реализовать простой чат,состоящий чат-сервера и чат-клиента. Сервер должен поддерживать соединение сразу от нескольких клиентов.Для этого необходимо организовать учет активных клиентов, сохраняя необходимые сведения о клиентах в определенных структурах данных.

Обмен сообщениями между клиентами осуществляется через сервер. При получении сообщения от какого-либо клиента сервер должен продублировать его на своем экране и отправить каждому из подключенных клиентов полученное сообщение. При подключении нового клиента сервер оповещает каждого клиента о новом пользователе, посылая им его имя и IP-адрес.

Тестирование сервера провести в режиме удаленного доступа при одновременном подключении не менее двух клиентов.

## 5.5 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 отличие взаимодействия клиента с сервером при использовании протоколов 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.
9. Что такое потоки выполнения, для чего они нужны и как они работают?
10. Какие проблемы возникают при использовании потоков и как эти проблемы решаются ?
11. Приведите схему взаимодействия chat-сервера и chat-клиента.
12. Опишите классы модуля threading.
13. Что такое семафор, в чем его отличие от замка ?

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

**Основной**

1. Олифер В.Г., Олифер Н.А. Компьютерные сети. Принципы, технологии, протоколы: учебник для вузов. Юбилейное издание.– СПб: Питер, 2020.–1008c.
2. Таненбаум Э., Уэзеролл Д. Компьютерные сети. 5-е изд.–СПб:Питер, 2012.– 960 с.
3. Куроуз Дж., Росс К. Компьютерные сети. Многоуровневая архитектура Интернета. 2-е изд. –СПб.: Питер, 2004. – 765с.
4. Камер Дуглас. Компьютерные сети и Internet. Разработка приложений для Internet: Пер. с англ. –М.: Издательский дом «Вильямс», 2002.–640с.
5. Снейдер Й. Эффективное программирование TCP/IP. Библиотека программиста. – СПб: Питер, 2002. – 320 с.
6. Семёнов Ю.А. Телекоммуникационные технологии.–[http://citforum.ru/nets/ semenov/](http://citforum.ru/nets/%20semenov/)[Электронный ресурс], дата обращения 30.10.2022
7. Новиков Ю. В. Основы локальных сетей: курс лекций: учебное пособие / Ю. В. Новиков, С. В. Кондратенко. — М. : Интернет-Университет Информационных Технологий (ИНТУИТ), 2005. — 360 c.

**Дополнительный**

1. Кобылянский В. Г. Сетевые информационные технологии. Моделирование и основные протоколы компьютерных сетей: учебное пособие. – Новосибирск: Изд-во НГТУ, 2021. –131 с.
2. Вся статистика Интернета и соцсетей на 2021 год - цифры и тренды в мире и в России. – <https://www.web-canape.ru/business/vsya-statistika-interneta-i-socsetej-na-2021-god-cifry-i-trendy-v-mire-i-v-rossii/>[Электронный ресурс], дата обращения 30.10.2022
3. Что такое и зачем нужен IPV6? Разбор.– [https://habr.com/ru/company/droider/blog/ 568778/](https://habr.com/ru/company/droider/blog/%20568778/)[Электронный ресурс], дата обращения 30.10.2022
4. GoogleIPv6.– <https://www.google.com/intl/en/ipv6/statistics.html>[Электронный ресурс], дата обращения 30.10.2022
5. Как настроить Хамачи. <http://hamachi-pc.ru/instructions/kak-nastroit-hamachi.html>[Электронный ресурс], дата обращения 30.10.2022]

КРАТКИЙ СЛОВАРЬ ТЕРМИНОВ

**Внешний (белый) IP-адрес**– IP-адрес, доступный из любой внешней сети.

**Демультиплексирование** – прием агрегированного потока информации, разделение его на отдельные потоки и передача потоков на компьютеры пользователей.

**Домен имен** –совокупность имен, у которых старшие составные части символьного имени совпадают.

**Звезда** – топология сети, при которой все сетевые узлы подключены к одному центральному узлу, в качестве которого могут использоваться коммуникационные устройства (концентратор или коммутатор) или отдельный управляющий компьютер (сервер).

**Инкапсуляция**– процесс объединения заголовков и данных с последующей их передачей на нижележащий уровень.

**Информационно-телекоммуникационнаясеть –** технологическая система,предназначеннаядляпередачиполиниямсвязиинформации,доступккоторойосуществляетсясиспользованиемсредстввычислительнойтехники.

**Коллизия** – ситуация, возникающая в сетях с разделяемой средой, когда две рабочие станции одновременно начинают передавать данные. При этом передаваемые кадры могут быть потеряны.

**Кольцо** –топология сети, основанная на последовательном соединении всех сетевых узлов, причем последний узел соединён с первым.

**Коммуникационная сеть**– это совокупность коммуникационногооборудования и линий связи, предназначенных для организации обмена информацией, представленной в требуемой форме.

**Коммуникационный домен** – параметр транспортного протокола, который задает систему адресации соединения.

**Коммутатор** (Switch) – многопроцессорное устройство канального уровня, имеющее несколько портов, которое при получении кадра на любой порт передает его только на порт, к которому подключен получатель.

**Коммутация** – процесс взаимодействия устройств сети для продвижения данных от отправителя адресату через транзитные узлы.

**Коммутируемая сеть** – сеть, в которой все узлы подключаются к коммутатору индивидуальными линиями связи.В этом случае каждый сетевой узел и соответствующий порт коммутатора взаимодействуют по схеме точка-точка.

**Контрольная сумма** – это целое число, значение которого зависит от значения каждого бита данных, при изменении хотя бы одного бита значение контрольной суммы изменяется. Используется для обнаружения ошибок в передаваемых данных.

**Концентратор** (Hub) – устройство физического уровня, имеющее несколько портов, которое при получении сигнала на любой порт дублирует его на все остальные порты.

**Локальная сеть** – компьютерная сеть, которая соединяет компьютеры в пределах ограниченной территории.

**Локальный (серый) IP-адрес–**IP-адрес, доступный только внутри локальной сети.

**Маршрутизатор** (Router) –устройство сетевого уровня, имеющее несколько портов, каждый из которых подключается к отдельной сети, используется для продвижения пакетов данных между сетями.

**Маршрутизация** – глобальный, охватывающий всю сеть, процесс определения пути, который проходит пакет данных от отправителя до получателя.

**Маска сети** –32-битное число, выделяющее в IP-адресе адрес сети и адрес узла: биты маски, соответствующие адресу сети, установлены в 1, остальные биты соответствуют адресу устройства в этой сети и имеют значение 0.

**Метрика –** целое число,определяющее критерий выбора маршрута передачи IP-пакета маршрутизатором при наличии нескольких альтернативных вариантов.

**Межсетевой экран** (**firewall, брандмауэр**) - это устройство, реализующее контроль за поступающей в локальную сеть и исходящей из нее информацией и обеспечивающие защиту локальной сети посредством фильтрации трафика.

**Мост** (Bridge) – однопроцессорное устройство канального уровня, имеющее несколько портов, которое при получении сигналов на любой порт передает его только на порт, к которому подключен получатель.

**Мультиплексирование** – объединение потоков информации, поступающей с компьютеров пользователей в один агрегированный поток и передача этого потока в коммутатор магистральной сети.

**Общая шина** – топология сети, основанная на параллельном подключении всех сетевых узлов к одной шине передачи данных.

**Повторитель** (Repeater) –устройство физического уровня, имеющее два порта, предназначенное для усиления и восстановления формы входных сигналов и их передачи на выходной порт.

**Продвижение данных** –локальные действия конкретного коммутатора или маршрутизатора по перемещению данных из входного порта в выходной порт.

**Пропускная способность** (Мбит/с) – количество пользовательских данных, переданных в единицу времени через все порты коммутатора, максимальное значение производительности достигается на кадрах максимальной длины.

**Рабочая станция –** компьютер, являющийся конечной точкой сетевого соединения, на котором пользователи сети выполняют свои прикладные задачи.

**Разделяемая среда** – среда передачи данных, к которой могут обращаться все сетевые узлы.

**Сегмент** – протокольная единица данных транспортного протокола TCP. **Сокет** – библиотека функций, обеспечивающих заданные параметры сеанса обмена данными и управляющих передачей и приемом данных. С логической точки зрения является абстрактным объектом, представляющим точку подключения приложения к сети.

**Сервер** – компьютер**,**предназначенный для хранения больших объемов данных и управления распределением сетевых ресурсов общего доступа.

**Сетевая карта (сетевой адаптер, сетевой интерфейс)** –контроллер, расположенный на материнской плате компьютера, предназначенный для приема сигналов из сети и передачи сигналов в сеть.

**Сетевой адрес–** уникальный адрес сетевого интерфейса, который используется для доступа к узлу из внешних сетей, имеет длину 4 байта для протокола IPv4 и 16 байтов для протокола IPv6.

**Скорость продвижения кадров** (кадров в секунду)– скорость, с которой в коммутаторе выполняются операции приема кадра в буфер, просмотр адресной таблицы и передача кадра в сеть через найденный в таблице порт.

**Стек протоколов**– иерархически организованный набор протоколов, достаточный для организации взаимодействия узлов в сети.

**Таблица коммутации** (**адресная таблица**) –таблица во внутренней памяти коммутатора (моста), в которой указывается соответствие MAC-адреса узла определенному порту коммутатора (моста).

**Топология сети –** конфигурация графа, вершинам которого соответствуют терминальные устройства и коммутационное оборудование, а ребрам – физические или информационные связи между вершинами.

**Тройное рукопожатие** – процедура установления логического соединения транспортным проколом TCP, включает три этапа: запрос на соединение от клиента, подтверждение от сервера, подтверждение от клиента.

**Хост** – компьютер, подключенный к компьютерной сети.

**Шлюз** (Gateway) – устройство сетевого уровня, имеющее несколько портов, каждый из которых подключается к отдельной сети, используется для продвижения пакетов данных между сетями, использующими разные протоколы (например, проводными и беспроводными).

**Элементарный канал** – базовая техническая характеристика, определяющая фиксированное значение пропускной способности канала в телефонных сетях и равная 64 Кбит/с.

**ARP –** протокол канального уровня, используемый для организации ассоциативной связи между IP-адресами и MAC- адресами.

**MAC-адрес –** уникальный локальный адрес сетевого интерфейса, используется в локальных сетях и имеет длину 6 байтов.

**DNS** – протокол прикладного уровня, используемый для организации ассоциативной связи между доменными именами и IP-адресами.

**DNS-сервер** – сетевой узел, на котором хранятся таблицы соответствия доменных имен и их IP-адресов.

**Ethernet** – семейство стандартов и технологий канального уровня, используемых в локальных сетях.

**ICMP** – протокол межсетевых управляющих сообщений, предназначенный для формирования и передачи сообщений об ошибках, возникающих во время прохождения IP-пакета по сети, и состоянии узла назначения IP-пакета.

**LLC** – подуровень канального уровня,который выполняет формирование кадров, реализует функции инкапсуляции и декапсуляции, а также взаимодействует с сетевым уровнем.

**TCP –** протокол транспортного уровня, обеспечивающий надежную доставку данных с предварительным установлением логического соединения между сетевыми узлами.

**TTL** *–* целое число, которое задаетвремя жизни IP-пакета. Интерпретируется как максимальное число транзитных узлов, которое разрешено пройти пакету на этапе доставки получателю,при прохождении каждого узла значение TTL уменьшается на единицу и при достижении нулевого значения пакет уничтожается.

**RIP** – протокол адаптивной маршрутизации, используемый в небольших сетях.

**UDP –** протокол транспортного уровня, реализующий дейтаграммный способ доставки данных.

# ПРИЛОЖЕНИЕ 1

Примерпростого удаленного взаимодействия

**Файлserver.cpp**// сервер

#pragmacomment (lib,"Ws2\_32.lib")

#include<winsock2.h>

#include<stdio.h>

#include<iostream>

#include<sstream>

#include<string>

usingnamespace std;

int main(void)

{

WORD sockVer;

WSADATA wsaData;

int retVal;

sockVer = MAKEWORD(2,2);

WSAStartup(sockVer, &wsaData);

SOCKET servSock = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);//Создаемсокет

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(2022);

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)

{

retVal = listen(servSock, 10); //Переводимсокетнапрослушивание if(retVal == SOCKET\_ERROR)

{

printf("Unable to listen\n");

WSACleanup();

system("pause");

return SOCKET\_ERROR;

}

SOCKET clientSock; //Ждемклиента

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);

if(retVal == SOCKET\_ERROR) //Принимаем данные от клиента

{

printf("Unable to recv\n");

system("pause");

return SOCKET\_ERROR;

}

printf("Data received\n");

string s = (constchar\*) szReq;

if(s[0]=='s') // Если команда на выключение сервера, то закрываем сокет

{

char \*szResp = "Server shutdown";

retVal = send(clientSock, szResp, 256, 0);

closesocket(clientSock);

break;

}

else

{

istringstreamss(s); // Выполняемобработкувходныхданных

int a, b;

ss >> a;

ss >> b;

char szResp[256];

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

printf("Sending response from server\n");//Отправляемрезультатклиенту

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");

}

}

closesocket(servSock); //Закрываемсокет

WSACleanup();

return 0;

}

**Файлclient.cpp**

// клиент

#pragmacomment (lib,"Ws2\_32.lib")

#include<stdio.h>

#include<winsock2.h>

#include<string>

#include<iostream>

usingnamespace std;

int main()

{

WORD ver = MAKEWORD(2,2);

WSADATA wsaData;

int retVal=0;

WSAStartup(ver,(LPWSADATA)&wsaData);

SOCKET clientSock = socket(AF\_INET, SOCK\_STREAM, IPPROTO\_TCP);//Создаемсокет

if(clientSock == SOCKET\_ERROR)

{

printf("Unable to create socket\n");

WSACleanup();

system("pause");

return 1;

}

string ip;

cout <<"Enter ip>";

cin >> ip;

cin.ignore();

SOCKADDR\_IN serverInfo;

serverInfo.sin\_family = AF\_INET;

serverInfo.sin\_addr.S\_un.S\_addr = inet\_addr(ip.c\_str());

serverInfo.sin\_port = htons(2022);

// подключаемсяксерверупо 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");

retVal = send(clientSock, pBuf, strlen(pBuf), 0); //отправляемданныенасервер

if(retVal == SOCKET\_ERROR)

{

printf("Unable to send\n");

WSACleanup();

system("pause");

return 1;

}

charszResponse[256]; // получаемответотсервера

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;

}