ФЕДЕРАЛЬНОЕ АГЕНСТВО ПО ОБРАЗОВАНИЮ

ГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ВЫСШЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
«САМАРСКИЙ ГОСУДАРСТВЕННЫЙ АЭРОКОСМИЧЕСКИЙ
УНИВЕРСИТЕТ имени академика С.П. КОРОЛЕВА»

ИЗУЧЕНИЕ СТЕКА ПРОТОКОЛОВ TCP/IP

Методические указания к лабораторной работе

САМАРА
Издательство СГАУ
2006

Составитель *И.В. Лофицкий*

УДК 681.324(075)

Изучение стека протоколов TCP/IP: метод. указания к лабораторной работе / сост. *И.В. Лофицкий.* – Самара: Изд-во Самар. гос. аэрокосм. ун-та, 2006. 48 с.

Приведены основные функции протоколов семейства TCP/IP. Рассмотрены принципы установки TCP-соединения и определения IP- и MAC-адреса локального и удаленных хостов.

Предназначены для студентов специальности 210302 «Радиотехника» и 200401 «Биотехнические и медицинские аппараты и системы». Подготовлены на кафедре радиотехники и МДС.

Печатаются по решению Редакционно-издательского совета Самарского государственного аэрокосмического университета

Рецензент В. В. Иванов

Цель работы: Изучение принципов построения и функционирования локальных вычислительных сетей Ethernet.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1. СТЕКИ СЕТЕВЫХ ПРОТОКОЛОВ

1.1. Семиуровневая модель OSI

Модель OSI (Open System Interconnect Reference Model, Эталонная модель взаимодействия открытых систем) представляет собой универсальный стандарт взаимодействия двух систем (компьютеров) через вычислительную сеть.

Эта модель описывает функции семи иерархических уровней и интерфейсы взаимодействия между уровнями. Каждый уровень определяется сервисом, который он предоставляет вышестоящему уровню, и протоколом – набором правил и форматов данных для взаимодействия между собой объектов одного уровня, работающих на разных компьютерах.

Идея состоит в том, что вся сложная процедура сетевого взаимодействия может быть разбита на некоторое количество простых процедур, последовательно выполняющихся объектами, соотнесенными с уровнями модели. Модель построена так, что объекты одного уровня двух взаимодействующих компьютеров сообщаются непосредственно друг с другом с помощью соответствующих протоколов, не зная, какие уровни лежат под ними и какие функции они выполняют. Задача объектов – предоставить через стандартизованный интерфейс определенный сервис вышестоящему уровню, воспользовавшись, если нужно, сервисом, который представляет данному объекту нижележащий уровень.

Например, некий процесс отправляет данные через сеть процессу, находящемуся на другом компьютере. Через стандартизован-

ный интерфейс процесс-отправитель передает данные нижнему уровню, который предоставляет процессу сервис по пересылке данных, а процесс-получатель через такой же стандартизованный интерфейс получает эти данные от нижнего уровня. При этом ни один из процессов не знает и не имеет необходимости знать, как именно осуществляет передачу данных протокол нижнего уровня, сколько еще уровней находится под ним, какова физическая среда передачи данных и каким путем они движутся.

Возможна также взаимозаменяемость объектов одного уровня (например, при изменении способа реализации сервиса) таким образом, что объект вышестоящего уровня не заметит подмены.

Объекты, выполняющие функции уровней, могут быть реализованы в программном, программно-аппаратном или аппаратном виде. Как правило, чем ниже уровень, тем больше доля аппаратной части в его реализации.

Организация сетевого взаимодействия компьютеров, построенного на основе иерархических уровней, называется протокольным стеком.

1.1.1. Уровни моделей OSI

Ниже перечислены (в направлении сверху вниз) уровни модели OSI и указаны их общие функции.

Уровень приложения (Application) – интерфейс с прикладными процессами.

Уровень представления (Presentation) – согласование представления (форматов, кодировок) данных прикладных процессов.

Сеансовый уровень (Session) – установление, поддержка и закрытие логического сеанса связи между удаленными процессами.

Транспортный уровень (Transport) – обеспечение безошибочного сквозного обмена потоками данных между процессами во время сеанса.

Сетевой уровень (Network) – фрагментация и сборка передаваемых транспортным уровнем данных, маршрутизация и продвижение их по сети от компьютера-отправителя к компьютеру-получателю.

Канальный уровень (Data Link) – управление каналом передачи данных, управление доступом к среде передачи, передача данных по каналу, обнаружение ошибок в канале и их коррекция.

Физический уровень (Physical) – физический интерфейс с каналом передачи данных, представление данных в виде физических сигналов и их кодирование (модуляция).

1.1.2. Инкапсуляция и обработка пакетов

При продвижении пакета сверху вниз каждый новый уровень добавляет к пакету свою служебную информацию в виде заголовка и, возможно, трейлера (информации, помещаемой в конец сообщения). Эта операция называется инкапсуляцией данных верхнего уровня в пакете нижнего уровня. Служебная информация предназначается для объекта того же уровня на удаленном компьютере, ее формат и интерпретация определяются протоколом данного уровня. Данные, приходящие с верхнего уровня, могут представлять собой пакеты с уже инкапсулированными данными еще более верхнего уровня.

При получении пакета от нижнего уровня он разделяется на заголовок (трейлер) и данные. Служебная информация из заголовка (трейлера) анализируется, и в соответствии с ней данные, возможно, направляются одному из объектов верхнего уровня. Тот, в свою очередь, рассматривает эти данные как пакет со своей служебной информацией и данными для еще более верхнего уровня, и процедура повторяется, пока пользовательские данные, очищенные от всей служебной информации, не достигнут прикладного процесса.

Возможно, что пакет данных не будет доведен до самого верхнего уровня, например, в случае, если данный компьютер представляет собой промежуточную станцию на пути между отправителем и получателем. В этом случае объект соответствующего уровня при анализе служебной информации заметит, что пакет на этом уровне адресован не ему (хотя с точки зрения нижележащих уровней он был адресован именно этому компьютеру). Тогда компьютер выполнит необходимые действия для перенаправления пакета к месту назначения или возврата отправителю с сообщением об ошибке, но в любом случае не будет продвигать данные на верхний уровень.

Модель OSI предложена достаточно давно, однако протоколы, на ней основанные, используются редко, во-первых, в силу своей не всегда оправданной сложности, во вторых, из-за существования хотя и не соответствующих строго модели OSI, но уже хорошо зарекомендовавших себя стеков протоколов TCP/IP.

1.2. Стек протокола TCP/IP

TCP/IP – собирательное название для набора (стека) сетевых протоколов разных уровней, используемых в Internet. Эти протоколы в терминологии модели OSI относятся к сетевому и транспортному уровням соответственно. IP обеспечивает продвижение пакета по составной сети, а TCP гарантирует надежность его доставки.

Особенности стека протокола TCP/IP:

- открытые стандарты протоколов, разрабатываемые независимо от программного и аппаратного обеспечения;
- независимость от физической среды передачи;
- система уникальной адресации;
- стандартизованные протоколы высокого уровня для распространенных пользовательских сервисов.

Стек протоколов TCP/IP делится на 4 уровня: *прикладной (application), транспортный (transport), межсетевой (internet) и уровень доступа к среде передачи (network access).*

Как и в модели OSI, данные более верхних уровней инкапсулируются в пакеты нижних уровней (рис. 1)

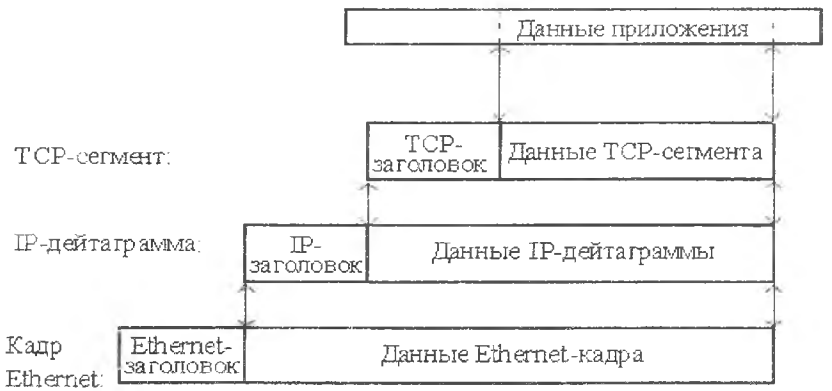


Рис. 1. Пример инкапсуляции пакетов в стеке

Примерное соотношение уровней стеков OSI и TCP/IP показано на рис. 2.

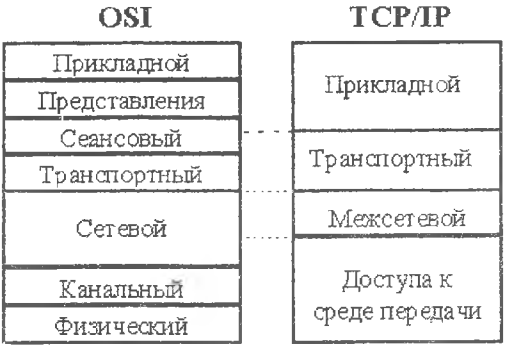


Рис. 2. Соотношение уровней стеков OSI и TCP/IP

Ниже рассмотрены функции каждого уровня и примеры протоколов.

1.2.1. Уровень приложений

Приложения, работающие со стеком TCP/IP, могут также выполнять функции уровней представления и частично сеансового модели OSI, например, преобразование данных к внешнему представлению, группировка данных для передачи и т.п.

Распространенными примерами приложений являются программы telnet, ftp, HTTP-серверы и клиенты (WWW-броузеры), программы работы с электронной почтой.

Для пересылки данных другому приложению приложение обращается к тому или иному модулю транспортного уровня.

1.2.2. Транспортный уровень

Протоколы транспортного уровня обеспечивают прозрачную (сквозную) доставку данных (end-to-end delivery service) между двумя прикладными процессами. Процесс, получающий или отправляющий данные с помощью транспортного уровня, идентифицируется на этом уровне номером, который называется *номером порта*. Таким образом, роль адреса отправителя и получателя на транспортном уровне выполняет номер порта (или проще – порт).

Анализируя заголовок своего пакета, полученного от межсетевого уровня, транспортный модуль определяет номер порта получателя, которому из прикладных процессов направлены данные, и пе-

редает эти данные соответствующему прикладному процессу (возможно, после проверки их на наличие ошибок и т.п.). Номера портов получателя и отправителя записываются в заголовок транспортным модулем, отправляющим данные. Заголовок транспортного уровня содержит также и другую служебную информацию. Формат заголовка зависит от используемого транспортного протокола.

На транспортном уровне работают два основных протокола: UDP и TCP (рис.3).

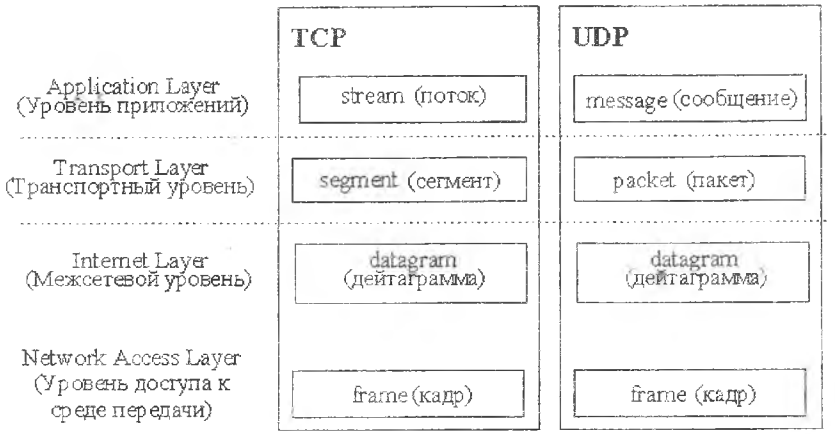


Рис. 3. Стек протоколов TCP/IP

TCP (Transmission Control Protocol – протокол контроля передачи) – надежный протокол с установлением соединения: он управляет логическим сеансом связи (устанавливает, поддерживает и закрывает соединение) между процессами и обеспечивает надежную (безошибочную и гарантированную) доставку прикладных данных от процесса к процессу.

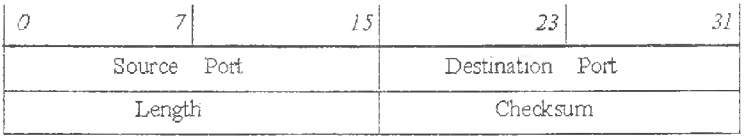
Данными для TCP является не интерпретируемая протоколом последовательность пользовательских октетов, разбиваемая для передачи по частям. Каждая часть передается в отдельном TCP-сегменте. Для продвижения сегмента по сети между компьютером-отправителем и компьютером-получателем модуль TCP пользуется сервисом межсетевого уровня (вызывает модуль IP).

UDP (User Datagram Protocol – протокол пользовательских дейтаграмм) используется либо при пересылке коротких сообщений, когда накладные расходы на установление сеанса и проверку успешной

доставки данных оказываются выше расходов на повторную (в случае неудачи) пересылку сообщения, либо в том случае, когда сама организация процесса-приложения обеспечивает установление соединения и проверку доставки пакетов.

Пользовательские данные, поступившие от прикладного уровня, предваряются UDP-заголовком, и сформированный таким образом UDP-пакет отправляется на межсетевой уровень.

UDP-заголовок состоит из двух 32-битных слов:



Значение полей:

- **Source Port** – номер порта процесса-отправителя;
- **Destination Port** – номер порта процесса получателя;
- **Length** – длина UDP-пакета вместе с заголовком;
- **Checksum** – контрольная сумма.

После заголовка непосредственно следуют пользовательские данные, переданные модулю UDP прикладным уровнем за один вызов. Протокол UDP рассматривает эти данные как целостное сообщение, он никогда не разбивает сообщение для передачи в нескольких пакетах и не объединяет несколько сообщений для пересылки в одном пакете.

При получении пакета от межсетевого уровня модуль UDP проверяет контрольную сумму и передает содержимое сообщения прикладному процессу, номер порта которого указан в поле " Destination Port ".

Если проверка контрольной суммы выявила ошибку, или если процесса, подключенного к требуемому порту, не существует, пакет игнорируется. Если пакеты поступают быстрее, чем модуль UDP успевает их обрабатывать, то поступающие пакеты также игнорируются. Протокол UDP не имеет никаких средств подтверждения безошибочного приема данных или сообщения об ошибке, не обеспечивает приход сообщений в порядке отправки, не производит предварительного установления сеанса связи между прикладными процессами, поэтому он является ненадежным протоколом без установления соединения. Если приложение нуждается в подобного рода

услугах, оно должно использовать на транспортном уровне протокол TCP.

Максимальная длина UDP-сообщения равна максимальной длине IP-дейтаграммы (65535 октетов) за вычетом минимального IP-заголовка (20) и UDP-заголовка (8), т.е. 65507 октетов. На практике обычно используются сообщения длиной 8192 октета.

Протокол UDP использует следующие прикладные процессы: NFS (Network File System – сетевая файловая система); TFTP (Trivial File Transfer Protocol – простой протокол передачи файлов); SNMP (Simple Network Management Protocol – простой протокол управления сетью); DNS (Domain Name Service – доменная служба имен).

1.2.3. Межсетевой уровень и протокол IP

Основным протоколом этого уровня является протокол IP (Internet Protocol).

Протокол IP доставляет блоки данных, называемых дейтаграммами, от одного IP-адреса к другому. IP-адрес является уникальным 32-битным идентификатором компьютера (точнее, его сетевого интерфейса). Данные для дейтаграммы передаются IP-модулю транспортным уровнем. IP-модуль предваряет эти данные заголовком, содержащим IP-адреса отправителя и получателя, а также другую служебную информацию. Сформированная таким образом дейтаграмма передается на уровень доступа к среде передачи (например, одному из физических интерфейсов) для отправки по каналу передачи данных.

Когда модуль IP получает дейтаграмму с нижнего уровня, он проверяет IP-адрес назначения. Если дейтаграмма адресована данному компьютеру, то данные из нее передаются на обработку модулю вышестоящего уровня (какому конкретно – указано в заголовке дейтаграммы). Если адрес назначения дейтаграммы – чужой, то модуль IP может принять два решения: первое – уничтожить дейтаграмму, второе – отправить ее дальше к месту назначения, определив маршрут следования, – так поступают промежуточные станции – маршрутизаторы.

На границе сетей с различными характеристиками может потребоваться разбить дейтаграмму на фрагменты, а потом собрать в единое целое на компьютере получателе. Эта тоже задача протокола IP.

Если модуль IP по какой-либо причине не может доставить дейтаграмму, она уничтожается. При этом модуль IP может отправить компьютеру-источнику этой дейтаграммы уведомление об ошибке. Такие уведомления отправляются с помощью протокола ICMP, являющегося неотъемлемой частью модуля IP. Более никаких средств контроля корректности данных, подтверждения их доставки, обеспечения правильного порядка следования дейтаграмм, предварительного установления соединения между компьютерами протокол IP не имеет. Эта задача возложена на транспортный уровень.

Многие IP-адреса имеют эквивалентную форму записи в виде доменного имени (например, IP-адрес 194.84.124.4 может быть записан как maria.vvsu.ru). Преобразование между этими двумя формами выполняется службой DNS. Доменные имена введены для удобства использования человеком. Все TCP/IP-процессы и коммуникационное оборудование используют только IP-адреса.

1.2.4. Уровень доступа к среде передачи

Функции этого уровня:

– отображение IP-адресов и физических адресов сети (MAC-адреса, например, Ethernet-адрес в случае сети Ethernet). Эту функцию выполняет протокол ARP;

– инкапсуляция IP-дейтаграмм в кадры для передачи по физическому каналу и извлечение дейтаграмм из кадров. При этом не требуется какого-либо контроля безошибочности передачи (хотя он может и присутствовать), поскольку в стеке TCP/IP такой контроль возложен на транспортный уровень или на само приложение. В заголовке кадров указывается точка доступа к сервису (SAP, Service Access Point) – поле, содержащее код протокола межсетевого уровня, которому следует передать содержимое кадра;

– определение метода доступа к среде передачи – т.е. способа, с помощью которого компьютер устанавливает свое право на произведение передачи данных;

– определение представления данных в физической среде;

– пересылка и прием кадра.

Стек TCP/IP не подразумевает использования каких-либо определенных протоколов уровня доступа к среде передачи и физических сред передачи данных. От уровня доступа к среде передачи требуется наличие интерфейса с модулем IP, обеспечивающего передачу дейтаграммы между уровнями. Также требуется обеспечить

преобразование IP-адреса узла сети, на который передается дейтаграмма, в MAC-адрес. Часто в качестве уровня доступа к среде передачи могут выступать целые протокольные стеки, тогда говорят об IP поверх ATM, IP поверх IPX, IP поверх X.25 и т.п.

2. ПРОТОКОЛ IP

2.1. Функции протокола IP

Протокол IP находится на межсетевом уровне стека протоколов TCP/IP. Функции протокола IP определены в стандарте RFC-791 следующим образом: «Протокол IP обеспечивает передачу блоков данных, называемых дейтаграммами, от отправителя к получателям, где отправители и получатели являются компьютерами, идентифицируемыми адресами фиксированной длины (IP-адресами). Протокол IP обеспечивает при необходимости также фрагментацию и сборку дейтаграмм для передачи данных через сети с малым размером пакетов».

Одна из основных задач, решаемых протоколом IP, – маршрутизация дейтаграмм, т.е. определение пути следования дейтаграммы от одного узла сети к другому на основании адреса получателя.

Порядок работы модуля IP на каком-либо узле сети, принимающего дейтаграмму из сети, следующий:

1) с одного из интерфейсов уровня доступа к среде передачи (например, с Ethernet-интерфейса) в модуль IP поступает дейтаграмма;

2) модуль IP анализирует заголовок дейтаграммы;

3) если пунктом назначения дейтаграммы является данный компьютер:

- то дейтаграмма является фрагментом большой дейтаграммы, ожидаются остальные фрагменты, после чего из них собирается исходная большая дейтаграмма;
- из дейтаграммы извлекаются данные и направляются на обработку одному из протоколов вышележащего уровня (какому именно, указывается в заголовке дейтаграммы);

4) если дейтаграмма не направлена ни на один из IP-адресов данного узла, то дальнейшие действия зависят от того, разрешена или запрещена ретрансляция (forwarding) “чужих” дейтаграмм;

5) если ретрансляция разрешена, то определяются следующий узел сети, на который должна быть переправлена дейтаграмма для

доставки ее по назначению, и интерфейс нижнего уровня, после чего дейтаграмма передается на нижний уровень этому интерфейсу для отправки;

6) если дейтаграмма ошибочна или по каким-либо причинам не может быть доставлена, она уничтожается, при этом отправителю дейтаграммы отсылается ICMP-сообщение об ошибке.

При получении данных от вышестоящего уровня для отправки их по сети IP-модуль формирует дейтаграмму с этими данными, в заголовок которой заносятся адреса отправителя и получателя (также полученные от транспортного уровня) и другая информация, после чего выполняются следующие шаги:

– если дейтаграмма предназначена этому же узлу, из нее извлекаются данные и направляются на обработку одному из протоколов транспортного уровня (какому именно указывается в заголовке дейтаграммы);

– если дейтаграмма не направлена ни на один из IP-адресов данного узла, то определяется следующий узел сети, на который должна быть направлена дейтаграмма для доставки ее по назначению, и интерфейс нижнего уровня, после чего дейтаграмма передается на нижний уровень этому интерфейсу для отправки, при необходимости может быть произведена фрагментация дейтаграммы;

– если дейтаграмма ошибочна или по каким-либо причинам не может быть доставлена, она уничтожается.

Узлом сети называется компьютер, подключенный к сети и поддерживающий протокол IP. Узел сети может иметь один и более IP-интерфейсов, подключенных к одной или разным сетям, каждый такой интерфейс идентифицируется уникальным IP-адресом.

IP-сетью называется множество компьютеров (IP-интерфейсов), часто, но не всегда подсоединенных к одному физическому каналу связи, способных пересылать IP-дейтаграммы друг другу непосредственно, при этом IP-адреса интерфейсов одной IP-сети имеют общую часть, которая называется адресом, или номером IP-сети, и специфическую для каждого интерфейса часть, называемую адресом или номером данного интерфейса в данной IP-сети.

Маршрутизатором, или **шлюзом**, называется узел сети с несколькими IP-интерфейсами, подключенными к разным IP-сетям, осуществляющий на основе решения задачи маршрутизации перенаправление дейтаграмм из одной сети в другую для доставки от отправителя к получателю.

Хостами называются узлы IP-сети, не являющиеся маршрутизаторами. Обычно хост имеет один IP-интерфейс, хотя может иметь и несколько.

Неотъемлемой частью IP-модуля является протокол ICMP (Internet Control Message Protocol), отправляющий диагностическое сообщение при невозможности доставки дейтаграммы и в ряде других случаев. Совместно с протоколом IP работает также протокол ARP (Address Resolution Protocol), выполняющий преобразование IP-адресов в MAC-адреса. Работа этих протоколов будет рассмотрена ниже.

2.2. IP-адреса

IP-адрес является уникальным 32-битным идентификатором IP-интерфейса в Internet. IP-адрес присваивается узлу сети в случае, если узел является хостом с одним IP-интерфейсом, в противном случае следует уточнять, об адресе какого именно интерфейса данного узла идет речь.

IP-адреса принято записывать разбивкой всего адреса по октетам, каждый октет записывается в виде десятичного числа, числа разделяются точками. Например адрес

10100000010100010000010110000011

записывается как

10100000.01010001.00000101.10000011=160.81.5.131.

IP-адрес хоста состоит из номера IP-сети, который занимает старшую область адреса, и номера хоста в этой сети, который занимает младшую часть. Положение границы сетевой и хостовой частей может быть различным, определяя различные типы IP-адресов.

2.2.1. Классовая модель

В классовой модели IP-адрес может принадлежать к одному из четырех классов сетей. Каждый класс характеризуется определенным размером сетевой части адреса, кратным восьми, таким образом, граница между сетевой и хостовой частями IP-адреса в классовой модели всегда проходит по границе октета. Принадлежность к тому или иному классу определяется по старшим битам адреса.

Класс A. Старший бит адреса равен нулю. Размер сетевой части равен 8 битам. Таким образом, может существовать всего примерно 2^7 сетей класса A, но каждая сеть обладает адресным пространством на 2^{24} хостов. Так как старший бит адреса нулевой, то все IP-адреса этого класса имеют значение старшего октета в диапазоне 0 – 127, который является также и номером сети.

Класс B. Два старших бита адреса равны 10. Размер сетевой части равен 16 битам. Таким образом, может существовать всего примерно 2^{14} сетей класса B, каждая сеть обладает адресным пространством на 2^{16} хостов. Значения старшего октета IP-адреса лежат в диапазоне 128 – 191, при этом номером сети являются два старших октета.

Класс C. Три старших бита адреса равны 110. Размер сетевой части равен 24 битам. Количество сетей класса C примерно 2^{21}, адресное пространство каждой сети рассчитано на 254 хоста. Значения старшего октета IP-адреса лежат в диапазоне 192 – 223, а номером сети являются три старших октета.

Класс D. Сети со значениями старшего октета IP-адреса 224 и выше зарезервированы для специальных целей. Некоторые адреса используются для мультикастинга – передачи дейтаграмм группе узлов сети, например:

224.0.0.1 – всем хостам данной сети;
224.0.0.2 – всем маршрутизаторам данной сети;
224.0.0.5 – всем OSPF-маршрутизаторам;
224.0.06 – всем выделенным OSPF-маршрутизаторам.

На рис. 4 представлены классы IP-адресов.

В классе A выделены две особые сети, их номера 0 и 127. Сеть 0 используется при маршрутизации как указание на маршрут по умолчанию и в других особых случаях.

IP- интерфейс с адресом сети 127 используется для адресации узлом себя самого (loopback, интерфейс обратной связи). Интерфейс обратной связи необязательно имеет адрес в сети 127 (особенно у маршрутизаторов), но если узел имеет IP-интерфейс с адресом 127.0.0.1, то это – интерфейс обратной связи. Обращение по адресу интерфейса обратной связи означает связь с самим собой (без выхода пакетов данных на уровень доступа к среде передачи). Для прото-

колов на уровнях транспортном и выше такое соединение неотличимо от соединения с удаленным узлом, что удобно использовать, например, для тестирования сетевого программного обеспечения.

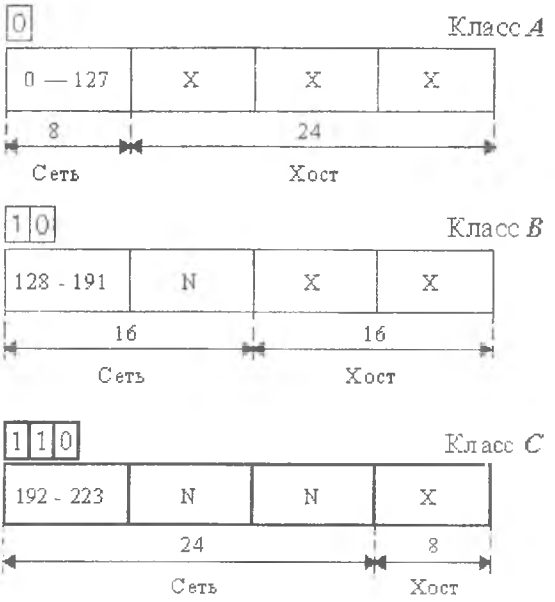


Рис.4. Классы IP-адресов

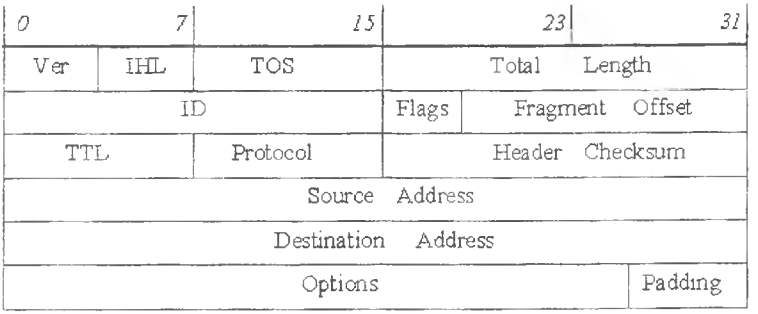
В любой сети все нули в номере хоста обозначают саму сеть, все единицы – адрес широковещательной передачи (broadcast).

Например, 194.124.84.0 – сеть класса С, номер хоста в ней определяется последним октетом. При отправлении широковещательного сообщения оно отправляется по адресу194.84.124.255. Номера, разрешенные для присвоения хостам: от 1 до 254 (194.84.124.1 – 194.84.124.254), всего 254 возможных адреса.

2.4. Формат заголовка IP-дейтаграммы

IP-дейтаграмма состоит из заголовка и данных. Заголовок состоит из 32-разрядных слов и имеет переменную длину, зависящую от размера поля “Options”, но всегда кратную 32 битам. За заголовком непосредственно следуют данные, передаваемые в дейтаграмме.

Формат заголовка:



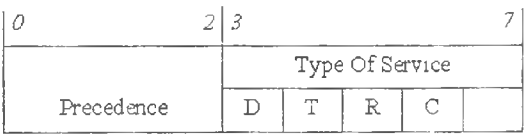
Значения полей заголовка следующие.

Ver (4 бита) – версия протокола IP, в настоящий момент используется версия 4, новые разработки имеют номера версий 6-8.

IHL (Internet Header Length) (4 бита) – длина заголовка в 32-битных словах; диапазон допустимых значений от 5 (минимальная длина заголовка, поле "Options" отсутствует) до 15 (т.е. может быть максимум 40 байт опций).

TOS (Type Of Service) (8 бит) – значение поля определяет приоритет дейтаграммы и желаемый тип маршрутизации.

Структура байта TOS следующая:



Три младших бита ("Precedence") определяют приоритет дейтаграммы:

111 – управление сетью;
110 – межсетевое управление;
101 – CRITIC-ECP;
100 – более чем мгновенно;
011 – мгновенно;
010 – немедленно;
001 – срочно;
000 – обычно.

Биты D, T, R, C определяют желаемый тип маршрутизации:

D (Delay) – выбор маршрута с минимальной задержкой;

T (Throughput) – выбор маршрута с максимальной пропускной способностью;

R (Reliability) – выбор маршрута с максимальной надежностью;

C (Cost) – выбор маршрута с минимальной стоимостью.

В дейтаграмме может быть установлен только один из битов D, T, R, C. Старший бит байта не используется.

Реальный учет приоритетов и выбор маршрута в соответствии со значением байта TOS зависят от маршрутизатора, его программного обеспечения и настроек. Маршрутизатор может поддерживать расчет маршрутов для всех типов TOS, для части или игнорировать TOS вообще. Маршрутизатор может учитывать значение приоритета при обработке дейтаграмм, исходящих только из некоторого ограниченного множества узлов сети, или вовсе игнорировать приоритет.

Total Length (16 бит) – длина всей дейтаграммы в октетах, включая заголовок и данные, максимальное значение 65535, минимальное – 21 (заголовок без опций и один октет в поле данных).

ID (Identification) (16 бит), **Flags** (3 бита), **Fragment Offset** (13 бит) используются для фрагментации и сборки дейтаграмм.

TTL (Time To Live) (8 бит) – "время жизни" дейтаграммы. Устанавливается отправителем, измеряется в секундах. Каждый маршрутизатор, через который проходит дейтаграмма, переписывает значение TTL, предварительно вычтя из него время, потраченное на обработку дейтаграммы. Поскольку скорость обработки данных на маршрутизаторах велика, на одну дейтаграмму тратится обычно меньше секунды, поэтому фактически каждый маршрутизатор вычитает из TTL единицу. При достижении значения TTL = 0 дейтаграмма уничтожается, при этом отправителю может быть послано соответствующее ICMP-сообщение. Контроль TTL предотвращает зацикливание дейтаграммы в сети.

Protocol (8 бит) – определяет программу (вышестоящий протокол стека), которой должны быть переданы данные дейтаграммы для дальнейшей обработки. Коды некоторых протоколов приведены в табл. 1.

Таблица 1

Коды IP-протоколов

Код	Протокол	Описание
1	ICMP	Протокол контрольных сообщений
2	IGMP	Протокол управления группой хостов
4	IP	IP поверх IP (инкапсуляция)
6	TCP	TCP
8	EGP	Протокол внешней маршрутизации (устарел)
9	IGP	Протокол внутренней маршрутизации (устарел)
17	UDP	UDP
46	RSVP	Протокол резервирования ресурсов при мультикастинге
88	IGRP	Протокол внутренней маршрутизации от фирмы cisco
89	OSPF	Протокол внутренней маршрутизации

Header Checksum (16 бит) – контрольная сумма заголовка, представляет из себя 16 бит, дополняющие биты в сумме всех 16-битовых слов заголовка. Перед вычислением контрольной суммы значение поля "Header Checksum" обнуляется. Поскольку маршрутизаторы изменяют значения некоторых полей заголовка при обработке дейтаграммы, контрольная сумма каждым маршрутизатором пересчитывается заново. Если при проверке контрольной суммы обнаруживается ошибка, дейтаграмма уничтожается.

Source Address (32 бита) – IP-адрес отправителя.

Destination Address (32 бита) - IP-адрес получателя.

Options – опции, поле переменной длины. Опций может быть одна, несколько или ни одной. Опции определяют дополнительные

услуги модуля IP по обработке дейтаграммы, в заголовок которой они включены.

Padding – выравнивание заголовка по границе 32-битного слова, если список опций занимает нецелое число 32-битных слов. Поле "Padding" заполняется нулями.

2.5. Протокол ICMP

Протокол ICMP (Internet Control Message Protocol, Протокол Управляющих Сообщений Интернет) является неотъемлемой частью IP-модуля. Он обеспечивает обратную связь в виде диагностических сообщений, посылаемых отправителю при невозможности доставки его дейтаграммы и в других случаях. ICMP стандартизован в RFC-792, дополнения – в RFC-950,256.

ICMP-сообщения не порождаются при невозможности доставки:

- дейтаграмм, содержащих ICMP-сообщения;
- не первых фрагментов дейтаграмм;
- дейтаграмм, направленных по групповому адресу (широковещание, мультикастинг);
- дейтаграмм, адрес отправителя которых нулевой или групповой.

Все ICMP-сообщения имеют IP-заголовок, значение поля "Protocol" равно 1. Данные дейтаграммы с ICMP-сообщением не передаются вверх по стеку протоколов для обработки, а обрабатываются IP-модулем.

После IP-заголовка следует 32-битное слово с полями "Тип", "Код" и "Контрольная сумма". Поля типа и кода определяют содержание ICMP-сообщения. Контрольная сумма считается так же, как и в IP-заголовке, но в этом случае суммируется содержимое ICMP-сообщения, включая поля "Тип" и "Код".

0 7	15	31
Тип	Код	Контрольная сумма

В табл. 2 представлены виды ICMP-сообщений.

Таблица 2

Виды ICMP-сообщений

Тип	Код	Сообщение
0	0	Echo Reply (эхо-ответ)
3		Destination Unreachable (адресат недостижим по различным причинам):
	0	Net Unreachable (сеть недоступна)
	1	Host Unreachable (хост недоступен)
	2	Protocol Unreachable (протокол недоступен)
	3	Port Unreachable (порт недоступен)
	4	DF=1 (необходима фрагментация, но она запрещена)
	5	Source Route failed (невозможно выполнить опцию Source Route)
4	0	Source Quench (замедление источника)
5		Redirect (выбрать другой маршрутизатор для посылки дейтаграмм)
	0	в данную сеть
	1	на данный хост
	2	в данную сеть с данным TOS
	3	на данный хост с данным TOS
8	0	Echo Request (эхо-запрос)
9	0	Router Advertisement (объявление маршрутизатора)
10	0	Router Solicitation (запрос объявления маршрутизатора)
11		Time Exceeded (время жизни дейтаграммы истекло)
	0	при передаче
	1	при сборке

Окончание табл. 2

12		Parameter problem (ошибка в параметрах)
	0	Ошибка в IP-заголовке
	1	Отсутствует необходимая опция
13	0	Timestamp (запрос временной метки)
14	0	Timestamp Reply (ответ на запрос временной метки)
17	0	Address Mask Request (запрос сетевой маски)
18	0	Address Mask Reply (ответ на запрос сетевой маски)

Ниже рассмотрены форматы ICMP-сообщений и даны комментарии к некоторым сообщениям.

Типы 3, 4, 11, 12

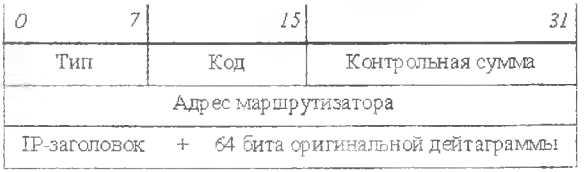
0 7	*15*	*31*
Тип	Код	Контрольная сумма
xxxxxxxxxx	не используется	
IP-заголовок +	64 бита оригинальной дейтаграммы	

В сообщении типа 12 в поле “xxxxxxxxxx” (1 октет) заносится номер октета заголовка, в котором обнаружена ошибка; в сообщениях типов 3, 4, 11 не используется. Все неиспользуемые поля заполняются нулями.

Сообщение типа 4 (“Замедление источника”) генерируется в случае переполнения (или опасности переполнения) буферов обработки дейтаграмм адресата или промежуточного узла на маршруте. При получении такого сообщения отправитель должен уменьшить скорость или приостановить отправку дейтаграмм до тех пор, пока он не перестанет получать сообщения этого типа.

IP-заголовок и начальные слова оригинальной дейтаграммы приводятся для опознания ее отправителем и, возможно, анализа причины сбоя.

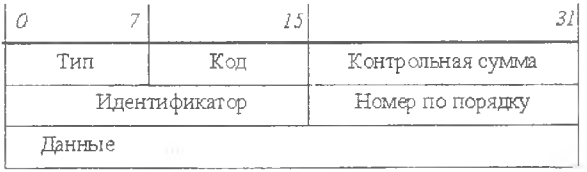
Тип 5



Сообщение типа 5 направляется маршрутизатором отправителю дейтаграммы в случае, когда маршрутизатор считает, что дейтаграммы в данное место назначения следует направлять через другой маршрутизатор. Адрес нового маршрутизатора приведен во втором слове сообщения.

Понятие "место назначения" конкретизируется значением поля "Код" (см. табл. 2). Информация о том, куда была направлена дейтаграмма, породившая ICMP-сообщение, извлекается из заголовка, присоединенного к сообщению. Отсутствие передачи сетевой маски ограничивает область применения сообщений типа 5.

Тип 0, 8

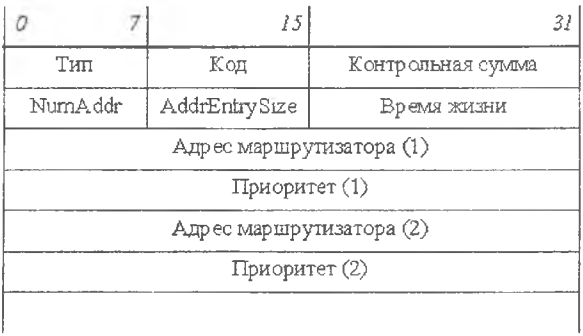


Сообщения типов 0 и 8 используются для тестирования сети по протоколу IP между двумя узлами сети. Тестирующий узел генерирует сообщения типа 8 ("Эхо-запрос"), при этом "Идентификатор" определяет данный сеанс тестирования (номер последовательности отправляемых сообщений). Поле "Номер по порядку" содержит номер данного сообщения внутри последовательности. В поле данных содержатся произвольные данные, размер i-го поля определяется общей длиной дейтаграммы, указанной в поле "Total length" IP-заголовка.

IP-модуль, получивший эхо-запрос, отправляет эхо-ответ. Для этого он меняет местами адреса отправителя и получателя, изменяет тип ICMP-сообщения на 0 и пересчитывает контрольную сумму.

Тестирующий узел по самому факту получения эхо-ответов, времени оборота дейтаграмм, проценту потерь и последовательности прибытия ответов может сделать выводы о наличии и качестве связи с тестируемым узлом. На основе посылки и приема эхо-сообщений работает программа **ping**.

Тип 9



Сообщения типа 9 (объявление маршрутизатора) периодически рассылаются маршрутизаторами хостам сети для того, чтобы хосты могли автоматически сконфигурировать свои маршрутные таблицы. Обычно такие сообщения рассылаются по мультикастинговому адресу 224.0.0.1 (“всем хостам”) или по широковещательному адресу.

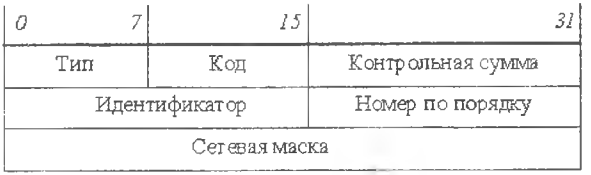
Сообщение содержит адреса одного или нескольких маршрутизаторов, снабженных значениями приоритета для каждого маршрутизатора. Приоритет является числом со знаком, записанным в дополнительном коде, – чем больше число, тем выше приоритет.

Поле “NumAddr” содержит количество адресов маршрутизаторов в данном сообщении; значение поля “AddrEntrySize” равно двум (размер поля, отведенного на информацию об одном маршрутизаторе, в 32-битных словах). “Время жизни” определяет срок годности информации, содержащейся в данном сообщении, в секундах.

Тип 10

Сообщения типа 10 (запрос объявления маршрутизатора) состоит из двух 32-битных слов, первое из которых содержит поля "Тип", "Код" и "Контрольная сумма", а второе зарезервировано (заполняется нулями).

Тип 17 и 18



Сообщения типов 17 и 18 (запрос и ответ на запрос значения маски сети) используются в случае, когда хост желает узнать маску сети, в которой он находится. Для этого в адрес маршрутизатора (или широковещательно, если адрес маршрутизатора неизвестен) отправляется запрос. Маршрутизатор отправляет в ответ сообщение с записанным в нем значением маски той сети, из которой пришел запрос. В том случае, когда отправитель запроса еще не знает своего IP-адреса, ответ отправляется широковещательно.

Поля "Идентификатор" и "Номер по порядку" могут использоваться для контроля соответствий запросов и ответов, но в большинстве случаев игнорируются.

3. ПРОТОКОЛ TCP

3.1. Функции протокола TCP

Протокол TCP (Transmission Control Protocol, Протокол контроля передачи) обеспечивает сквозную доставку данных между прикладными процессами, запущенными на узлах, взаимодействующих по сети. Стандартное описание TCP содержится в RFC-793.

TCP – надежный байт-ориентированный (byte-stream) протокол с установлением соединения. TCP находится на транспортном уров-

не стека TCP/IP, между протоколом IP и собственно приложением. Протокол IP занимается пересылкой дейтаграмм по сети, никак не гарантируя доставку, целостность, порядок прибытия информации и готовность получателя к приему данных, все эти задачи возложены на протокол TCP.

При получении дейтаграммы модуль IP передает данные этой дейтаграммы модулю TCP. Эти данные представляют собой TCP-сегмент, содержащий TCP-заголовок и данные пользователя (прикладного процесса). Модуль TCP анализирует служебную информацию заголовка, определяет, какому именно процессу предназначены данные пользователя, проверяет целостность и порядок прихода данных и подтверждает их прием другой стороной. По мере получения правильной последовательности неискаженных данных пользователя они передаются прикладному процессу.

3.1.1. Базовая передача данных

Модуль TCP выполняет передачу непрерывных потоков данных между своими клиентами в обоих направлениях. Клиентами TCP являются прикладные процессы, вызывающие модуль TCP при необходимости получить или отправить данные процессу-клиенту на другом узле.

Протокол TCP рассматривает данные клиента как непрерывный неинтерпретируемый поток октетов. TCP разделяет этот поток на части для пересылки на другой узел в TCP-сегментах некоторого размера. Для отправки или получения сегмента модуль TCP вызывает модуль IP.

Немедленное отправление данных может быть затребовано процессом-клиентом от TCP-модуля с помощью специальной функции PUSH, иначе TCP сам будет решать, как накапливать и когда отправлять данные клиента или когда передавать клиенту полученные данные.

3.1.2. Обеспечение достоверности

Модуль TCP обеспечивает защиту от повреждения, потери, дублирования и нарушения очередности получения данных.

Для выполнения этих задач все октеты в потоке данных сквозным образом пронумерованы в возрастающем порядке. Заголовок

каждого сегмента содержит число октетов данных в сегменте и порядковый номер первого октета той части потока данных, которая пересылается в данном сегменте. Например, если в сегменте пересылаются октеты с номерами от 2001 до 3000, то номер первого октета в данном сегменте равен 2001, а число октетов равно 1000.

Номер первого байта в потоке определяется на этапе установления соединения и обозначается ISN + 1. Например, ISN + 1 = 1.

Также для каждого сегмента вычисляется контрольная сумма, позволяющая обнаружить повреждение данных.

При удачном приеме октета данных принимающий модуль посылает отправителю подтверждение о приеме – номер удачно принятого октета. Если в течение некоторого времени отправитель не получит подтверждения, считается, что октет не дошел или был поврежден, и он посылается снова. Этот механизм контроля надежности называется PAR (Positive Acknowledgment with Retransmission). В действительности подтверждение посылается не для одного октета, а для некоторого числа последовательных октетов.

Нумерация октетов используется также для упорядочения данных в порядке очередности и обнаружения дубликатов (которые могут быть посланы из-за большой задержки при передаче подтверждения или потери подтверждения).

3.1.3. Разделение каналов

Протокол TCP обеспечивает работу одновременно нескольких соединений. Каждый прикладной процесс идентифицируется номером порта. Заголовок TCP-сегмента содержит номера портов процесса-отправителя и процесса-получателя. При получении сегмента модуль TCP анализирует номер порта получателя и отправляет данные соответствующему прикладному процессу.

Все распространенные сервисы Интернет имеют стандартизованные номера портов. Например, номер порта сервера электронной почты – 25, сервера FTP – 21. Список стандартных номеров портов можно найти в файле /etc/services (Unix).

Совокупность IP-адреса и номера порта называется сокетом. Сокет уникально идентифицирует прикладной процесс в Интернете. Например, сокет сервера электронной почты на хосте 194.84.124.4 обозначается как 194.84.124.4.25. Часто номер порта отделяется двоеточием.

3.1.4. Управление соединениями

Соединение – это совокупность информации о состоянии потока данных, включающая сокеты, номера посланных, принятых и подтвержденных октетов, размеры окон.

Каждое соединение уникально идентифицируется в Интернете парой сокетов.

Соединение характеризуется для клиента именем, которое является указателем на структуре TCB (Transmission Control Block), содержащим информацию о соединении.

Открытие соединения клиентом осуществляется вызовом функции OPEN, которой передается сокет, с которым требуется установить соединение. Функция возвращает имя соединения. Различают два типа открытия соединения: активное и пассивное.

При активном открытии TCP-модуль начинает процедуру установления соединения с указанным сокетом, при пассивном – ожидает, что удаленный TCP-модуль начнет процедуру установления соединения с указанного сокета. Указание 0.0.0.0.0 в качестве сокета при пассивном открытии означает, что ожидается соединение с любого сокета. Такой способ применяется в демонах – серверах Интернет, которые ждут установления соединения от клиента. Клиент же применяет процедуру активного открытия, сокет при этом формируется из IP-адреса сервера и стандартного номера порта для данного сервиса.

Закрытие соединения клиентом производится с помощью функции CLOSE, которой передается имя соединителя.

Процедура установления соединения происходит следующим образом.

Предположим, узел А желает установить соединение с узлом В. Первый отправляемый из А в В TCP-сегмент не содержит полезных данных, а служит для установления соединения. В его заголовке (в поле Flags) установлен бит SYN, означающий запрос связи, и содержится ISN (Initial Sequence Number – начальный номер последовательности) – число, начиная с которого узел А будет нумеровать отправляемые октеты (например, 0). В ответ на получение такого сегмента узел В откликается посылкой TCP-сегмента, в заголовке которого установлен бит ACK, подтверждающий установление соединения для получения данных от узла А. Так как протокол TCP обеспечивает полнодуплексную передачу данных, то узел В в этом же сегменте устанавливает бит SYN, означающий запрос связи для

передачи данных от B к A, и передает свой ISN (например, 0). Полезных данных этот сегмент также не содержит. Третий TCP-сегмент в сеансе посылается из A в B в ответ на сегмент, полученный из B. Так как соединение A – B можно считать установленным (получено подтверждение от B), то узел A включает в свой сегмент полезные данные, нумерация которых начинается с номера ISN(A)+1. Данные нумеруются по количеству отправленных октетов. В заголовке этого же сегмента узел A устанавливает бит ACK, подтверждающий установление связи B – A, что позволяет хосту B включить в свой следующий сегмент полезные данные для A. Последовательность установления TCP соединения иллюстрируется рис. 5.

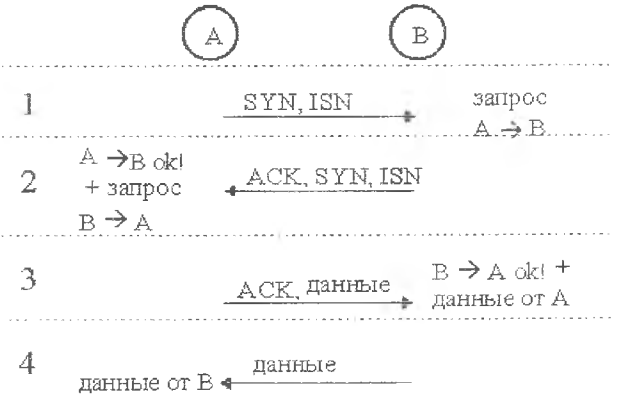


Рис.5. Установление TCP-соединения

Сеанс обмена данными заканчивается процедурой разрыва соединения, которая аналогична процедуре установления, с той разницей, что вместо SYN для разрыва используется служебный бит FIN ("данных для отправки больше не имею"), который устанавливается в заголовке последнего сегмента с данными, отправляемого узлом.

3.1.5. Управление потоком

Для ускорения и оптимизации процесса передачи больших объемов данных протокол TCP определяет метод управления потоком, называемый методом скользящего окна, который позволяет отправителю посылать очередной сегмент, не дожидаясь подтверждения о получении в пункте назначения предшествующего сегмента.

Протокол TCP формирует подтверждения не для каждого конкретного успешно полученного пакета, а для всех данных от начала посылки до некоторого порядкового номера ACK SN (Acknowledge Sequence Number) включительно. В качестве подтверждения успешного приема, например, первых 2000 байт, высылается ACK SN = 2001: это означает, что все данные в байтовом потоке под номерами от ISN + 1 = 1 до данного ACK SN – 1 (2000) успешно получены.

Вместе с посылкой отправителю ACK SN получатель объявляет также "размер окна", например – 6000 (рис.6). Это значит, что отправитель может посылать данные с порядковыми номерами от текущего ACK SN = 2001 до (ACK SN + размер окна – 1) = 8000, не дожидаясь подтверждения со стороны получателя.

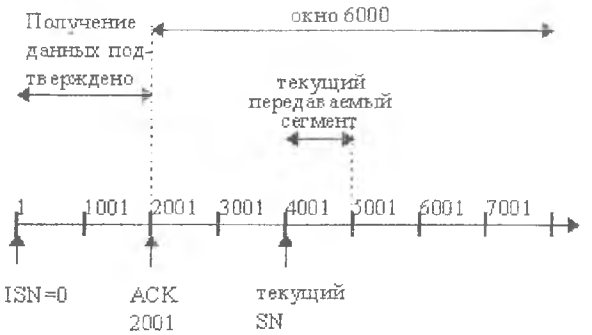


Рис. 6. Метод скользящего окна

Допустим, в данный момент отправитель посылает тысячеоктетный сегмент с порядковым номером данных SN = 4001. Если не будет получено новое подтверждение (новый ACK SN), отправитель будет посылать данные, пока он остается в пределах объявленного окна, т. е. до номера 8001. После этого посылка данных будет прекращена до получения очередного подтверждения и (возможно) нового размера окна. Однако размер окна выбирается таким образом, чтобы подтверждения успевали приходить вовремя и остановки передачи не происходило – для этого и предназначен метод скользящего окна. Размер окна может динамически изменяться получателем.

Для временной остановки посылки данных достаточно объявить нулевое окно. Но даже и в этом случае через определенные

промежутки времени будут отправляться сегменты с одним октетом данных. Это делается для того, чтобы отправитель гарантированно узнал о том, что получатель вновь объявил ненулевое окно, поскольку получатель обязан подтвердить получение "пробных" сегментов, а в этих подтверждениях он укажет также и текущий размер своего окна.

Протокол TCP позволяет вести полнодуплексную передачу. Один и тот же сегмент, высылаемый, например, из B в A, может содержать в заголовке служебную информацию по подтверждению полученных данных от A, а в поле данных – полезные данные для A.

Модуль TCP может использовать алгоритм "медленного старта", формируя при установлении соединения окно перегрузки, размер которого изначально равен размеру одного сегмента. Это окно показывает, сколько сегментов TCP-модуль, с его собственной точки зрения, может отправить без получения подтверждения. Скользящее же окно показывает, какой объем неподтвержденных данных модулю разрешено отправить с точки зрения удаленного модуля, получателя его данных. После прихода подтверждения от получателя окно перегрузки увеличится на 1 сегмент, и отправитель может выслать уже два сегмента, не дожидаясь подтверждения. Такой подход позволяет постепенно увеличивать нагрузку на сеть. Если окно перегрузки становится больше скользящего окна, объявляемого получателем, ограничение на передачу неподтвержденных данных устанавливает уже скользящее окно получателя.

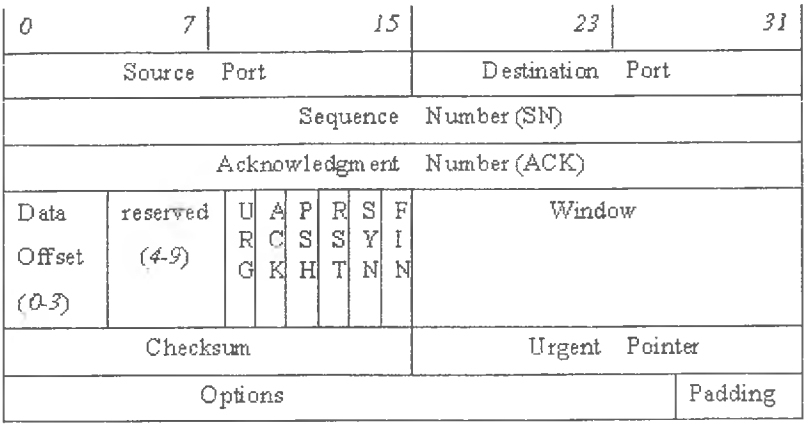
В случае, если никакие данные приложения не передаются, а соединение открыто, модуль TCP может периодически посылать сегменты-зонды для выяснения того, не отключилась ли другая сторона без уведомления партнера (например, в результате обрыва линии или другим некорректным образом). Такое зондирование проводится примерно каждые два часа неактивности.

3.2. Заголовок TCP-сегмента

TCP-сегмент состоит из заголовка и данных.

Заголовок сегмента состоит из 32-разрядных слов и имеет переменную длину, зависящую от размера поля Options, но всегда кратную 32 битам. За заголовком непосредственно следуют данные – часть потока данных пользователя, передаваемая в данном сегменте.

Формат заголовка:



Значения полей заголовка следующие:

Source Port (16 бит), **Destination Port** (16 бит) – номера портов процесса- отправителя и процесса-получателя соответственно.

Sequence Number (SN) (32 бита) – порядковый номер первого октета в поле данных сегмента среди всех октетов потока данных для текущего соединения, т. е. если в сегменте пересылаются октеты с 2001-го по 3000-й, то SN = 2001. Если в заголовке сегмента установлен бит SYN (фаза установления соединения), то в поле SN записывается начальный номер (ISN), например 0. Номер первого октета данных, посылаемых после завершения фазы установления соединения, равен ISN+1.

Acknowledgment Number (ACK) (32 бита) – если установлен бит ACK, то это поле содержит порядковый номер октета, который отправитель данного сегмента желает получить. Это означает, что все предыдущие октеты (с номерами от ISN + 1 до ACK – 1 включительно) были успешно получены.

Data Offset (4 бита) – длина TCP-заголовка в 32-битных словах.

Reserved (6 бит) – зарезервировано; заполняется нулями.

Control Bits (6 бит) – управляющие биты; активным является положение “бит установлен”.

URG – поле срочного указателя (Urgent Pointer) задействовано.

ACK – поле номера подтверждения (Acknowledgment Number) задействовано.

PSH – осуществить "проталкивание" – если модуль TCP получает сегмент с установленным флагом PSH, то он немедленно передает все данные из буфера приема процессу-получателю для обработки, даже если буфер не был заполнен.

RST – перезагрузка текущего соединения.

SYN – запрос на установление соединения.

FIN – нет больше данных для передачи.

Window (16 бит) – размер окна в октетах.

Checksum (16 бит) – контрольная сумма, представляет собой 16 бит, дополняющие биты в сумме всех 16-битовых слов сегмента (само поле контрольной суммы перед вычислением обнуляется). Контрольная сумма, кроме заголовка сегмента и поля данных, учитывает 96 бит псевдозаголовка, который для внутреннего употребления ставится перед TCP-заголовком. Этот псевдозаголовок содержит IP-адрес отправителя (4 октета), IP-адрес получателя (4 октета), нулевой октет, 8-битовое поле "Протокол", аналогичное полю в IP-заголовке, и 16 бит TCP-сегмента, измеренных в октетах. Такой подход обеспечивает защиту протокола TCP от ошибшихся в маршруте сегментов. Информация для псевдозаголовка передается через интерфейс "Протокол TCP/межсетевой уровень" в качестве аргументов или результатов запросов от протокола TCP к протоколу IP.

Urgent Pointer (16 бит) – используется для указания длины срочных данных, которые размещаются в начале поля данных сегмента. Указывает смещение октета, следующего за срочными данными, относительно первого октета в сегменте. Например, в сегменте передаются октеты с 2001-го по 3000-й, при этом первые 100 октетов являются срочными данными, тогда Urgent Pointer = 100. Протокол TCP не определяет, как именно должны обрабатываться срочные данные, но предполагает, что прикладной процесс будет предпринимать усилия для их быстрой обработки. Поле Urgent Pointer задействовано, если установлен флаг URG.

Options – поле переменной длины, может отсутствовать или содержать одну опцию или список опций, реализующих дополнительные услуги протокола TCP. Опция состоит из октета "Тип опций", за которым могут следовать октет "Длина опции в октетах" и октеты с данными для опции.

Стандарт протокола TCP определяет три опции (типы 0, 1, 2).

Опции типов 0 и 1 ("Конец списка опций" и "Нет операции" соответственно) состоят из одного октета, содержащего значение типа опции. При обнаружении в списке опции "Конец списка опций" разбор опций прекращается, даже если длина заголовка сегмента (Data Offset) еще не исчерпана. Опция "Нет операции" может использоваться для выравнивания между опциями по границе 32 бит.

Опция типа 2 "Максимальный размер сегмента" состоит из 4 октетов: одного октета типа опции (значение равно 2), одного октета длины (значение равно 4) и двух октетов, содержащих максимальный размер сегмента, который способен получать TCP-модуль, отправивший сегмент с данной опцией. Опцию следует использовать в SYN-сегментах на этапе установления соединения.

Padding – выравнивание заголовка по границе 32-битного слова, если список опций занимает нецелое число 32-битных слов. Поле Padding заполняется нулями.

3.3. Промежуточное состояние соединения

TCP-соединение во время функционирования проходит через ряд промежуточных состояний. Это состояния LISTEN, SYN-SENT, SYN-RECEIVED, ESTABLISHED, FIN-WAIT-1, FIN-WAIT-2, CLOSE-WAIT, CLOSING, LAST-ACK, TAIM-WAIT, а также фиксированное состояние CLOSED (состояние CLOSED является фиктивным, поскольку оно представляет отсутствие соединения). Переход из одного состояния в другое происходит в ответ на события, такие как запросы клиента, приход сегментов, истечение контрольного времени.

Определены следующие запросы процесса-клиента модулю TCP (с каждым запросом, кроме OPEN, передается имя соединения):

ACTIVE-OPEN – активное открытие соединения;

PASSIVE-OPEN – пассивное открытие соединения;

SEND – отправка данных (передается указатель на буфер данных, размер буфера, значение флагов URG и PSH);

RECEIVE – получение данных (передается указатель на буфер данных, размер буфера, возвращается счетчик полученных октетов, значение флагов URG и PSH);

STATUS – запрос состояния соединения;

CLOSE - закрытие соединения (производится досылка всех неотправленных данных и обмен сегментами с битом FIN);

ABORT – ликвидация соединения (уничтожаются блок TCB и все неотправленные данные, посылается сегмент с битом RST).

Деятельность программы протокола TCP можно рассматривать как реагирование на события в зависимости от состояния соединения. Состояния соединения следующие:

LISTEN – процесс пассивно ждет запроса со стороны чужих сокетов;

SYN-SENT – процесс отправил свой SYN и ждет чужого SYN;

SYN-RECEIVED – процесс получил чужой SYN, отправил (раньше или только что) свой SYN и ждет ACK на свой SYN;

ESTABLISHED – процесс отправил ACK на чужой SYN, получил ACK на свой SYN, соединение установлено;

FIN-WAIT-1 – процесс первый отправил свой FIN и ждет реакцию той стороны, при этом он, возможно, продолжает получать данные;

FIN-WAIT-2 – процесс получил ACK на свой ранее отправленный FIN, но не получил чужой FIN, ждет чужой FIN, при этом, возможно, продолжает получать данные;

CLOSE-WAIT – процесс, не отправив свой FIN (возможно, не собираясь прекращать соединение), получает чужой FIN, он отправляет ACK на чужой FIN, но при этом, возможно, продолжает отправлять данные;

LAST-ACK – процесс отправил свой FIN, но ранее он уже получил FIN с той стороны и отправил на него ACK на свой FIN для окончательного закрытия соединения;

CLOSING – процесс ранее отправил свой FIN и еще не получил не него подтверждение, но получил чужой FIN (и отправил на него ACK); ждет ACK на свой FIN;

TIME-WAIT – процесс ранее отправил свой FIN и получил на него подтверждение, получил чужой FIN и только что отправил на него ACK; теперь процесс ждет некоторое время (два времени жизни сегмента, обычно 4 минуты) для гарантии того, что та сторона получит его ACK на свой FIN, после чего соединение будет окончательно закрыто;

CLOSED – соединение отсутствует.

Диаграммы фазы установления соединения и фазы закрытия соединения представлены на рис. 7 и 8 соответственно.

Проблемы возникновения некорректных ситуаций, например, наполовину открытое соединение, получение заблудившихся в сети старых SYN- сегментов, неожиданный крах программ и т. д., решаются путем детектирования ошибки (несоответствие или бессмысленные значения ACK или SN), после чего посылается сигнал RST (сегмент с установленным битом RST) и соединение ликвидируется.

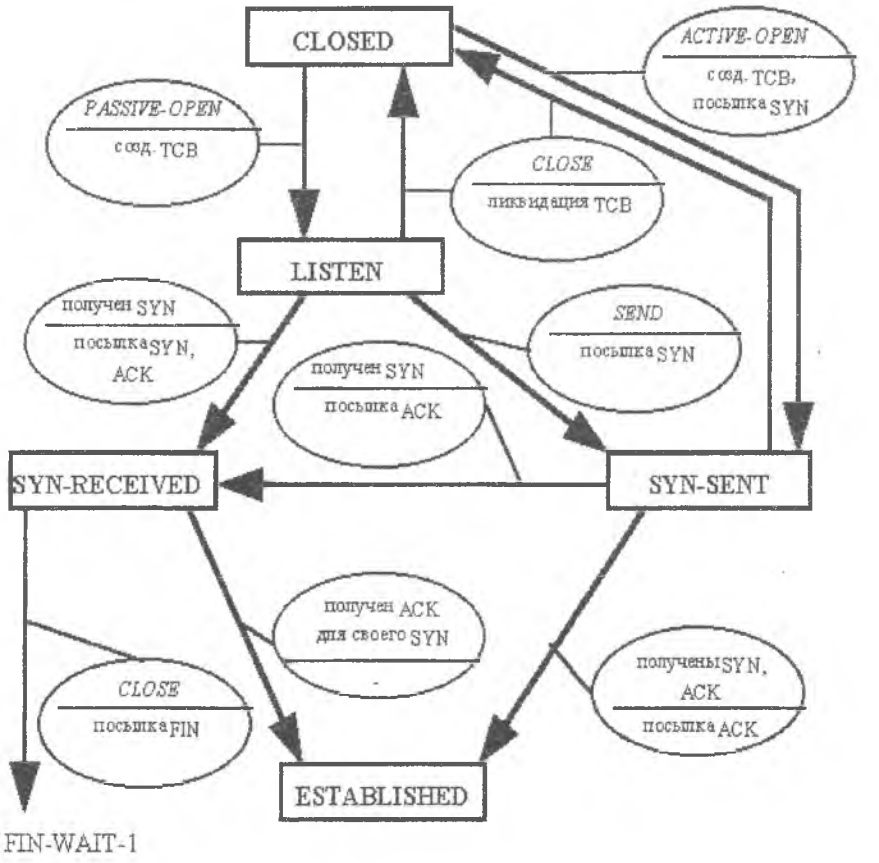


Рис. 7. Фаза установления соединения

Рис. 8. Фаза закрытия соединения

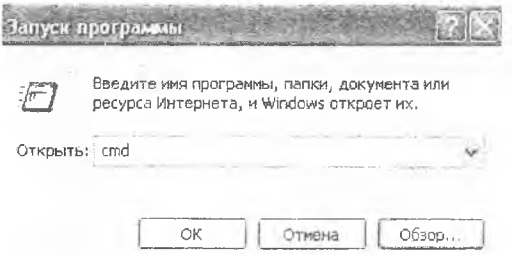
ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Определение IP- и MAC-адреса локального и удаленных хостов.
2. Изучение структуры ICMP-пакета.
3. Изучение принципа установления TCP-соединения.
4. Выявление широковещательных пакетов и определение их назначения.

Определение IP- и MAC-адреса локального и удаленных хостов

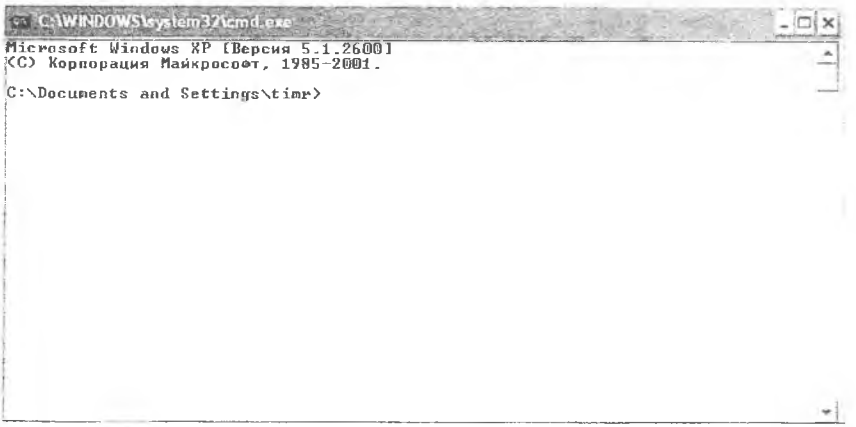
Для определения IP- и MAC-адреса локального хоста применяется команда ipconfig.

Эта команда выполняется из командной строки.



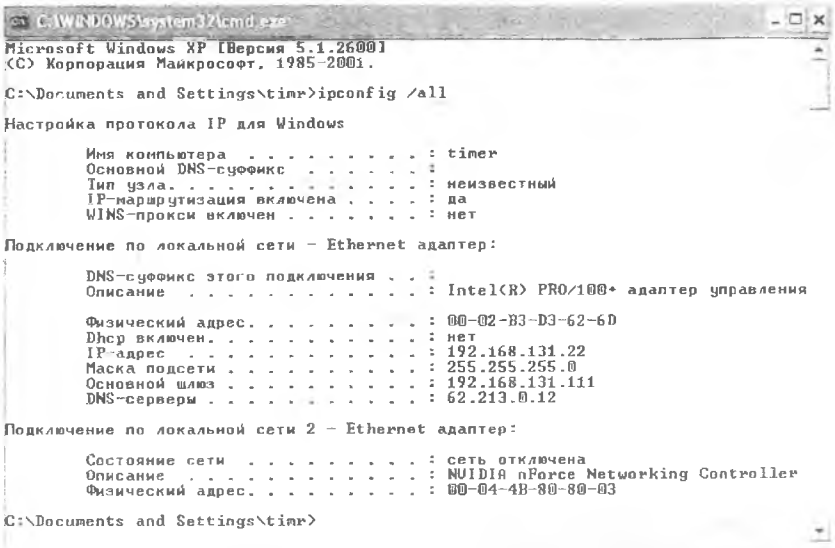
Пуск → Выполнить → «cmd» → Ok.

Откроется консоль Windows:



Для отображения полной информации о сетевых подключениях необходимо выполнить команду ipconfig/all.

Определите IP- и MAC-адреса по полученной информации.



Для определения состояния удаленного хоста применяется утилита Ping. Она посылает служебные запросы (ICMP-пакеты), результатом которых является наличие ответа на это сообщение, а также время отклика. Следует заметить, что отсутствие отклика удаленной системы еще не говорит о том, что она отключена от локальной сети.

Утилиту Ping удобнее всего использовать в консоле cmd.

Узнайте у преподавателя IP-адрес тестируемого удаленного хоста.

Использование: ping [-t] [-a] [-n число] [-l размер] [-f] [-i TTL] [-v TOS] [-r число]
[-s число] [[-j список узлов] | [-k список узлов]] [-w тайм аут] конечное имя.

Параметры:

– t	Отправка пакетов на указанный узел до команды прерывания. Для вывода статистики и продолжения нажмите <Ctrl>+<Break>, для прекращения – <Ctrl>+<C>
– a	Определение адресов по именам узлов
– n число	Число отправляемых запросов

– l размер	Размер буфера отправки
– f	Установка флага, запрещающего фрагментацию пакета
– i TTL	Задание срока жизни пакета (поле “Time To Live”)
– v TOS	Задание типа службы (поле “Type Of Service”)
– r число	Запись маршрута для указанного числа переходов
– s число	Штамп времени для указанного числа переходов
– j список узлов	Свободный выбор маршрута по списку узлов
– k список узлов	Жесткий выбор маршрута по списку узлов
– w тайм аут	Таймаут каждого ответа в миллисекундах

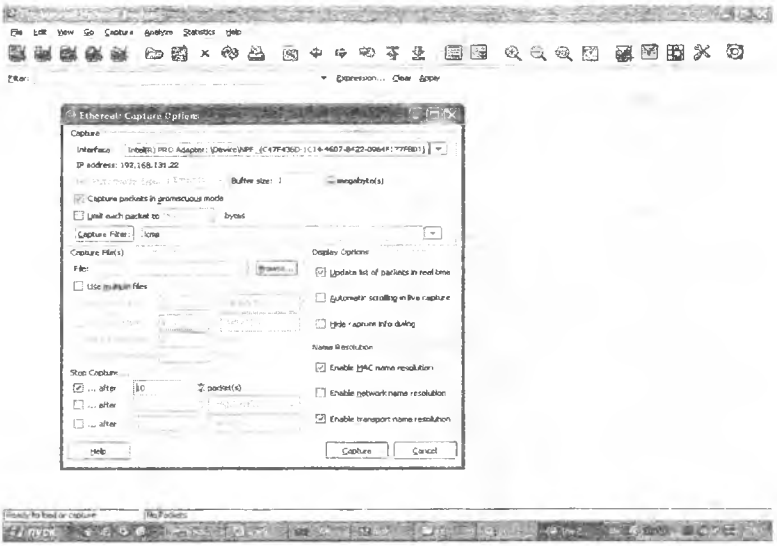
Запишите максимальное, минимальное и среднее время отклика.

Структура ICMP-пакета

Для изучения структуры ICMP-пакета используйте программу Ethereal. Программа Ethereal отображает отправляемые и принимаемые пакеты.

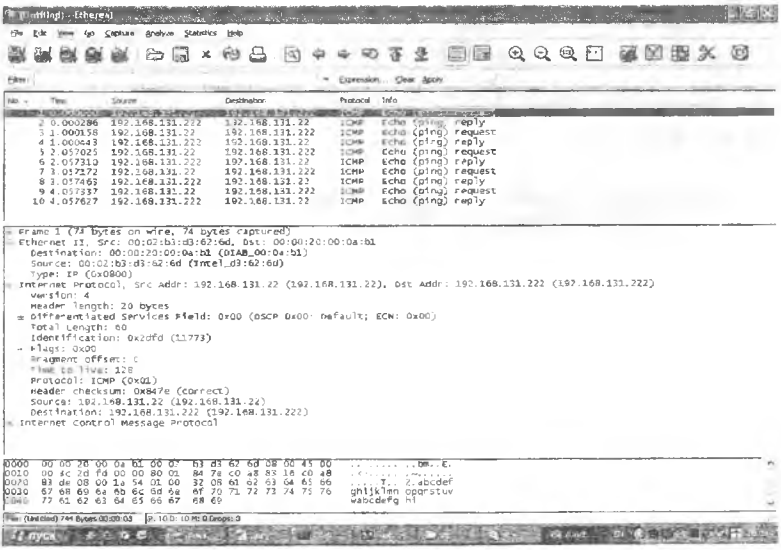
Для начала запустите программу ping с ключом t. Затем запустите программу Ethereal .

Нажмите на вторую слева кнопку “Show the capture options”.



Тип захватываемых пакетов – ICMP.

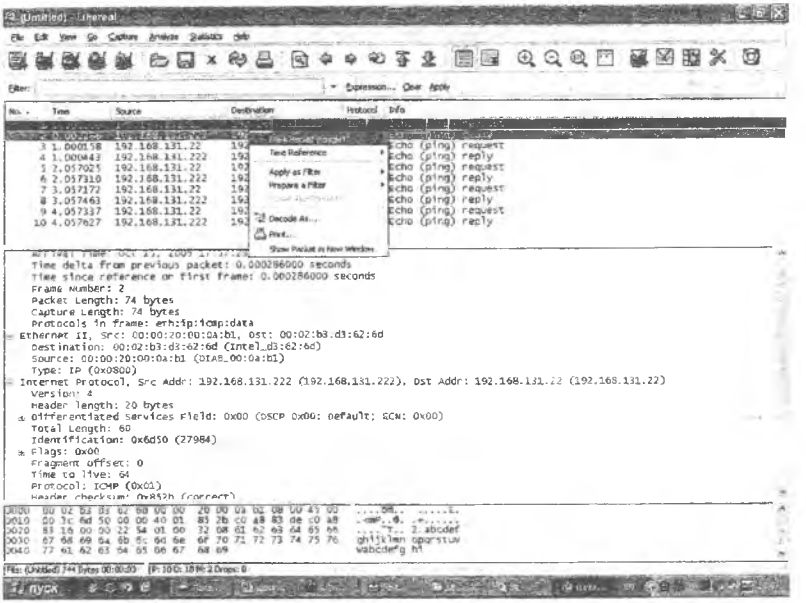
Остановите процесс захвата после 10 пакетов. Затем нажмите кнопку “Capture”.



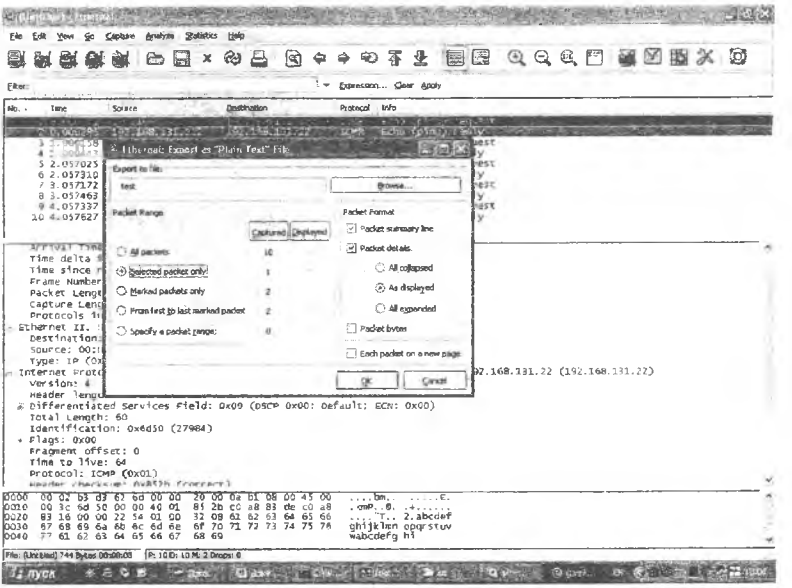
На экране появятся ICMP-пакеты.

Определите MAC-адрес удаленного хоста, а также определите время отклика системы.

Экспортируйте данные любого запросного и ответного ICMP-пакета. Для этого выделите 2 пакета.

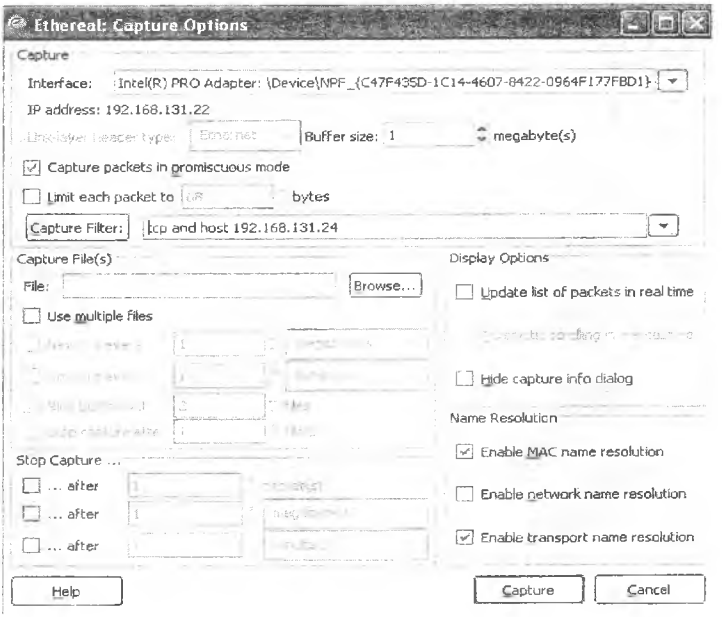


File - > Export -> Plain text.



TCP-соединение

Укажите в фильтре следующие параметры: tcp and host <IP>, где <IP> удаленный хост, выполняющий роль сервера TCP-соединений.



После запуска сниффера в адресной строке эксплорера напишите \\<IP>.

Найдите в полученных пакетах пакеты SYN, ASK SYN (RST, FIN), и сохраните их в отдельный файл.

Широковещательные сообщения

Укажите в фильтре следующие параметры: udp, поставьте галочку на Update list of packets in real time. Запустите сниффер. Скопируйте один широковещательный пакет в текстовый файл.

СОДЕРЖАНИЕ ОТЧЕТА

1. Цель работы.
2. Листинг ICMP-, TCP- и Broadcast-пакетов.
3. Выводы по работе.

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Какие функции выполняет протокол IP?
   - осуществляет негарантированную передачу пакетов между узлами;
   - осуществляет гарантированную передачу пакетов между узлами;
   - отвечает за адресацию узлов;
   - обеспечивает надежность соединения и отслеживает ошибки передачи;
   - ни одну из перечисленных функций.
2. Какие функции выполняет протокол TCP?
   - осуществляет негарантированную передачу пакетов между узлами;
   - осуществляет гарантированную передачу пакетов между узлами;
   - отвечает за адресацию узлов;
   - обеспечивает надежность соединения и отслеживает ошибки передачи;
   - ни одну из перечисленных функций.
3. К какому классу сетей относится IP-адрес 212.193.11.100?
   - A;
   - B;
   - C;
   - D;
   - E.
4. На какое количество подсетей делит маска 255.255.255.240 подсеть класса C?
   - 2;
   - 6;
   - 30;
   - 14;
   - 15.
5. Укажите IP-адреса, допустимые при маске подсети 255.255.255.224.
   - 212.193.11.032;
   - 212.193.11.064;
   - 212.193.11.067;

– 212.193.11.196
– 212.193.11.225.

6. Какой из указанных доменов является доменом верхнего уровня?

– www;
– http://www;
– ru;
– http;
– нет правильного ответа.

7. Какой протокол уровня Internet отвечает за диагностику и сообщения об ошибках при неудачной доставке данных?

– ICMP;
– FTP;
– IP;
– ARP;
– IGMP.

8. Какие TCP-порты используют приложение FTP Server?

– 20, 21;
– 62, 53;
– 125, 62, 31;
– 80, 53;
– 20, 53.

9. MAC-адрес – это 12-значный шестнадцатеричный номер, который принадлежит:

– сетевой плате;
– модему;
– сетевому кабелю;
– пакету данных;
– нет правильного ответа.

10. Идентификатор узла, содержащий только числа 255, используется для:

– обозначения идентификатора локальной сети;
– обозначения идентификатора узла;
– идентификатора глобальной сети;
– широковещательной рассылки;
– замыкания на себя.

11. Какие протоколы семейства TCP/IP находятся на уровне Internet?

– ATM и Ethernet;

– HTTP и FTP;
– DNS и WINS;
– TCP и UDP;
– IP, ICMP, IGMP и ARP.

12. Какой протокол используется для интерактивной передачи файлов?
– IP;
– TCP;
– FTP;
– HTTP;
– ICMP.

13. Какой из представленных протоколов входит в стек протоколов TCP/IP?
– SPX;
– NetBEUI;
– IrDA;
– IPX;
– ни один из представленных.

14. Дан IP-адрес 184.241.18.132. Каким будет идентификатор широковещательной рассылки в подсети, к которой принадлежит этот адрес?
– 184.255.255.255;
– 184.241.255.0;
– 184.241.255.255;
– 184.241.18.255;
– 184.241.0.0.

15. Из каких компонентов состоит пакет данных?
– заголовок, данные, концевой блок;
– заголовок, данные;
– заголовок, данные, трейлер;
– трейлер, данные.

16. Какие протоколы семейства TCP/IP находятся на уровне сетевого интерфейса?
– DNS и WINS;
– IP, ICMP, IGMP и ARP;
– ATM и Ethernet;
– TCP и UDP;
– HTTP и FTP.

17. С помощью какого параметра программы Ping можно установить время жизни пакета (TTL)?

- TIME;
- T;
- VR;
- TTL;
- нет правильного ответа.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

2. Олифер В.Г., Олифер Н.А. Компьютерные сети. – СПб.: Питер, 2004.

3. Ричард Стивен. Протоколы TCP/IP. Практическое руководство : Пер. с англ. – М.: Лори, 2000.

4. Хант Крейг. TCP/IP. Сетевое администрирование: Пер. с англ. – 3-е изд. – Киев: ДиаСофт, 2002.

Учебное издание

ИЗУЧЕНИЕ СТЕКА ПРОТОКОЛОВ ТСР/IP

Методические указания к лабораторной работе

Составитель ***Лофицкий Игорь Вадимович***

Редактор Л. Я. Чегодаева
Компьютерная верстка Т. Е. Половнева

Подписано в печать 4.12.06 г. Формат 60x84 1/16.
Бумага офсетная. Печать офсетная.
Усл. печ. л. 2,8. Усл. кр.-отт. 2,9. Уч.-изд.л. 3,0.
Тираж 100 экз. Заказ 133 . Арт. С- 56/2006

Самарский государственный
аэрокосмический университет .
443086 Самара, Московское шоссе, 34.

Изд-во Самарского государственного
аэрокосмического университета.
443086 Самара, Московское шоссе, 34.