МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«Юго-Западный государственный университет»

(ЮЗГУ)

Кафедра информационной безопасности

УТВЕРЖДАЮ
Проректор по учебной работе
_____О.Г. Локтионова
«___»_____2017 г.

ПОНЯТИЕ СЕТЕВЫХ ПРОТОКОЛОВ. ПРОТОКОЛЫ TCP, UDP

Методические указания по выполнению лабораторной и практической работы по дисциплинам «Сети и системы передачи информации», «Безопасность систем и сетей передачи данных», «Сети и системы передачи информации (специальные разделы)», «Администрирование вычислительных сетей», «Администрирование защищенных телекоммуникационных систем» для студентов укрупненной группы специальностей и направлений подготовки 10.00.00.

Курск 2017

УДК 004
Составители: И.В. Калуцкий, А.Г. Спеваков, Е.В. Шеин, К.О. Хохлач.

Рецензент
Кандидат технических наук, доцент кафедры
«Информационная безопасность» *М.О. Таныгин*

Понятие сетевых протоколов. Протоколы TCP, UDP: методические указания к выполнению лабораторных и практических работ по дисциплинам / Юго-Зап. гос. Ун-т; сост. И.В. Калуцкий, А.Г. Спеваков, Е.В. Шеин, К.О. Хохлач. Курск, 2017, 23 с.: ил. 11.; Библиогр.: с. 23.

Содержат сведения по сетевым протоколам различных уровней (TCP и UDP). Указывается порядок выполнения лабораторных и практических работ, правила оформления, содержание отчета.

Методические указания по выполнению лабораторных и практических работ по дисциплинам «Безопасность систем и сетей передачи данных», «Сети и системы передачи информации», «Сети и системы передачи информации (специальные разделы)», «Администрирование вычислительных сетей», «Администрирование защищенных телекоммуникационных систем» для студентов укрупненной группы специальностей и направлений подготовки 10.00.00.

Текст печатается в авторской редакции

Подписано в печать Формат 60х84 1/16.
Усл. печ. л. 1,34. Уч. –изд.л. 1,21. Тираж 30 экз. Заказ . Бесплатно.
Юго-Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Обмен данными, ориентированный на соединения, может использовать надежную связь, для обеспечения которой протокол уровня 4 посылает подтверждения о получении данных и запрашивает повторную передачу, если данные не получены или искажены. Протокол TCP использует именно такую надежную связь. TCP используется в таких прикладных протоколах, как HTTP, FTP, SMTP и Telnet.

Протокол TCP требует, чтобы перед отправкой сообщения было открыто соединение. Серверное приложение должно выполнить так называемое пассивное открытие (passive open), чтобы создать соединение с известным номером порта, и, вместо того чтобы отправлять вызов в сеть, сервер переходит в ожидание поступления входящих запросов. Клиентское приложение должно выполнить активное открытие (active open), отправив серверному приложению синхронизирующий порядковый номер (SYN), идентифицирующий соединение. Клиентское приложение может использовать динамический номер порта в качестве локального порта.

В отличие от TCP UDP — очень быстрый протокол, поскольку в нем определен самый минимальный механизм, необходимый для передачи данных. Конечно, он имеет некоторые недостатки. Сообщения поступают в любом порядке, и то, которое отправлено первым, может быть получено последним. Доставка сообщений UDP вовсе не гарантируется, сообщение может потеряться, и могут быть получены две копии одного и того же сообщения. Последний случай возникает, если для отправки сообщений в один адрес использовать два разных маршрута.

UDP не требует открывать соединение, и данные могут быть отправлены сразу же, как только они подготовлены. UDP не отправляет подтверждающие сообщения, поэтому данные могут быть получены или потеряны. Если при использовании UDP требуется надежная передача данных, ее следует реализовать в протоколе более высокого уровня.

ЦЕЛЬ РАБОТЫ

Цель лабораторной работы – изучить понятие и назначение сетевых протоколов, изучить протоколы сетевого и транспортного уровней: IP, TCP, UDP.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Получить задание
2. Изучить теоретическую часть
3. Выполнить практическое задание
4. Написать вывод

СОДЕРЖАНИЕ ОТЧЕТА

1. Титульный лист
2. Задание в соответствии с вариантом
3. Выполненное задание
4. Вывод

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ

Понятие протоколов

Сетевой протокол в компьютерных сетях – набор правил для специфического типа связи.

Разные протоколы зачастую описывают лишь разные стороны одного типа связи; взятые вместе, они образуют стек протоколов. Названия «протокол» и «стек протоколов» также указывают на программное обеспечение, которым реализуется протокол.

Новые протоколы для Интернета определяются IETF (Internet Engineering Task Force – проблемная группа проектирования Internet), а прочие протоколы – IEEE или ISO. ITU-T (International Telecommunication Union, ITU) занимается телекоммуникационными протоколами и форматами.

Эталонная модель OSI

Чтобы помочь поставщикам в стандартизации и интеграции их сетевого программного обеспечения, международная организация по стандартизации (ISO) определила программную модель пересылки сообщений между компьютерами. Эта модель получила название *эталонной модели OSI* (Open Systems Interconnection). В ней определено семь уровней программного обеспечения (рис. 1).

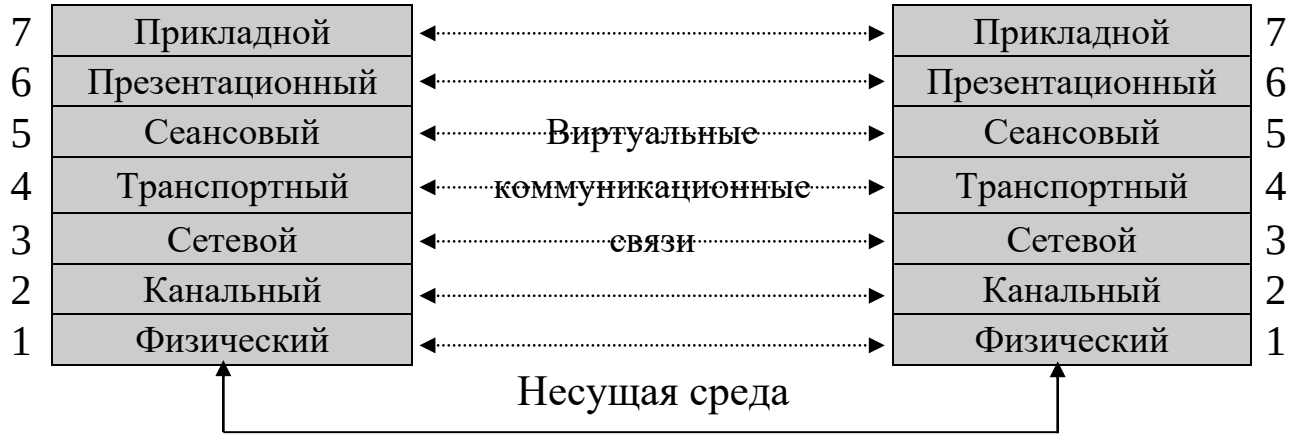

Рисунок 1 - Эталонная модель OSI.

Эталонная модель OSI – идеал, точно реализованный лишь в очень немногих системах, но часто используемый при объяснении основных принципов работы сети. Каждый уровень на одной из машин считает, что

он взаимодействует с тем же уровнем на другой машине. На данном уровне обе машины «разговаривают» на одном языке, или протоколе. Но в действительности сетевой запрос должен сначала пройти до самого нижнего уровня на первой машине, затем он передается по несущей среде и уже на второй машине вновь поднимается до уровня, который его поймет и обработает.

Задача каждого уровня в том, чтобы предоставить сервисы более высоким уровням и скрывать от них конкретную реализацию этих сервисов.

Прикладной уровень (Application layer). Верхний (7-й) уровень модели, обеспечивает взаимодействие сети и пользователя. Уровень разрешает доступ к сетевым службам приложениям пользователя, таким как обработчик запросов к базам данных, доступ к файлам, пересылке электронной почты. Также отвечает за передачу служебной информации, предоставляет приложениям информацию об ошибках и формирует запросы к уровню представления.

Уровень представления (Presentation layer). Этот уровень отвечает за преобразование протоколов и кодирование/декодирование данных. Запросы приложений, полученные с уровня приложений, он преобразует в формат для передачи по сети, а полученные из сети данные преобразует в формат, понятный приложениям. На этом уровне может осуществляться сжатие/распаковка или кодирование/раскодирование данных, а также перенаправление запросов другому сетевому ресурсу, если они не могут быть обработаны локально.

Сеансовый уровень (Session layer). Отвечает за поддержание сеанса связи, позволяя приложениям взаимодействовать между собой длительное время. Уровень управляет созданием/завершением сеанса, обменом информацией, синхронизации задач, определением права на передачу данных и поддержание сеанса в периоды неактивности приложений. Синхронизация передачи обеспечивается помещением в поток данных контрольных точек, начиная с которых возобновляется процесс при нарушении взаимодействия.

Транспортный уровень (Transport layer). 4-й уровень модели, предназначен для доставки данных без ошибок, потерь и дублирования в той последовательности, как они были переданы. При этом неважно какие данные передаются, откуда и куда, то есть он предоставляет сам механизм передачи. Блоки данных он разделяет на фрагменты, размер которых зависит от протокола, короткие объединяет в один, длинные

разбивает. Протоколы этого уровня предназначены для взаимодействия типа точка-точка.

Сетевой уровень (Network layer). 3-й уровень сетевой модели OSI, предназначен для определения пути передачи данных. Отвечает за трансляцию логических адресов и имён в физические, определение кратчайших маршрутов, коммутация и маршрутизация пакетов, отслеживание неполадок и заторов в сети. На этом уровне работает такое сетевое устройство, как маршрутизатор.

Канальный уровень (Data Link layer). Этот уровень предназначен для обеспечения взаимодействия сетей на физическом уровне и контроле за ошибками, которые могут возникнуть. Полученные данные от физического уровня он упаковывает в кадры данных, проверяет на целостность, если нужно исправляет ошибки и отправляет на сетевой уровень. Канальный уровень может взаимодействовать с одним или несколькими физическими уровнями, контролируя и управляя этим взаимодействием. Спецификация IEEE 802 разделяет этот уровень на 2 подуровня – MAC (Media Access Control) регулирует доступ к разделяемой физической среде, и LLC (Logical Link Control) обеспечивает обслуживание сетевого уровня. На этом уровне работают коммутаторы, мосты и сетевые адаптеры.

В программировании этот уровень представляет драйвер сетевой платы, в операционных системах имеется программный интерфейс взаимодействия канального и сетевого уровня между собой, это не новый уровень, а просто реализация модели для конкретной ОС. Примеры таких интерфейсов: ODI, NDIS.

Физический уровень (Physical layer). Самый нижний уровень модели, предназначен непосредственно для передачи потока данных. Осуществляет передачу электрических или оптических сигналов в кабель и соответственно их приём и преобразование в биты данных в соответствии с методами кодирования цифровых сигналов. Другими словами осуществляет интерфейс между сетевым носителем и сетевым устройством. На этом уровне работают концентраторы и повторители (ретрансляторы) сигнала.

Инкапсуляция и обработка пакетов

При продвижении пакета данных по уровням 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.

Разумеется, данные, приходящие с верхнего уровня, могут на самом деле представлять собой пакеты с уже инкапсулированными данными еще более верхнего уровня.

С другой стороны, 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 объект 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.

Применимость модели OSI

Хотя модель OSI полезна как основа для обсуждения сетевых архитектур и реализаций, ее нельзя рассматривать как готовый чертеж для создания любой сетевой архитектуры. Не следует также думать, что размещение некоторой функции на уровне N в этой модели означает, что только здесь наилучшее для нее место.

Модель OSI имеет множество недостатков. Хотя, в конечном итоге, были созданы работающие реализации, протоколы OSI на сегодняшний день утратили актуальность. Основные проблемы этой модели в том, что, во-первых, распределение функций между уровнями произвольно и не всегда очевидно, во-вторых, она была спроектирована (комитетом) без готовой реализации.

Другая проблема модели OSI – это сложность и неэффективность. Некоторые функции выполняются сразу на нескольких уровнях. Так,

обнаружение и исправление ошибок происходит на большинстве уровней.

Наконец, выбор именно семи уровней продиктован, скорее, политическими, а не техническими причинами. В действительности, сеансовый уровень и уровень представления редко встречаются в реально работающих сетях.

Стек протоколов TCP/IP

TCP/IP – собирательное название для набора (стека) сетевых протоколов разных уровней, используемых в Интернет. Особенности 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

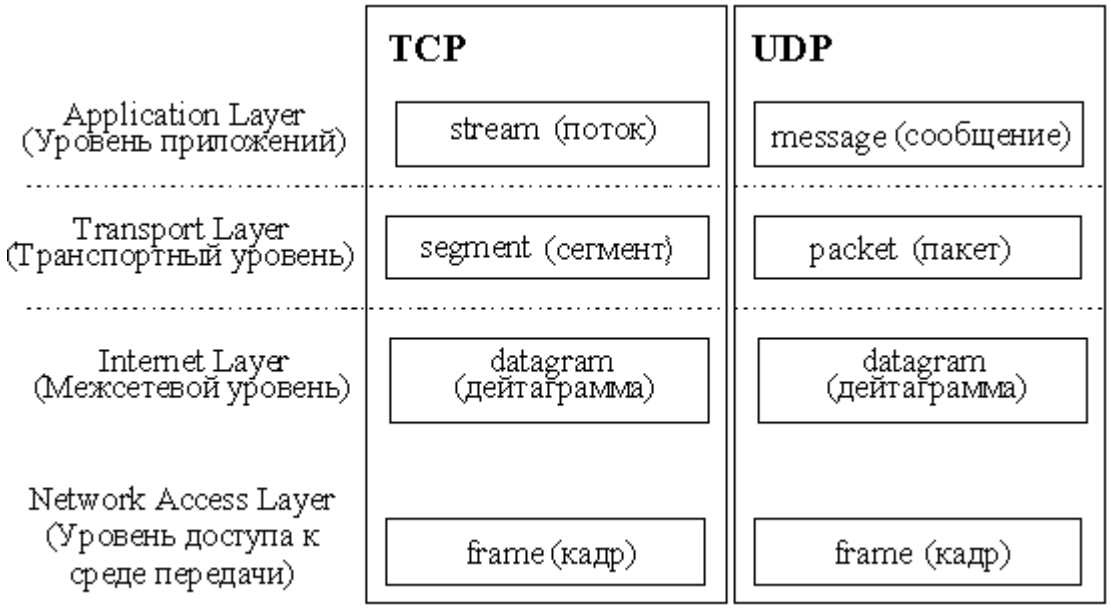

Рисунок 2 - Уровни стека протоколов TCP/IP.

Стек протоколов TCP/IP делится на 4 уровня: прикладной (application), транспортный (transport), межсетевой (internet) и уровень доступа к среде передачи (network access). Термины, применяемые для обозначения блока передаваемых данных, различны при использовании разных протоколов транспортного уровня – TCP и UDP, поэтому на

рисунке 2 изображено два стека. Как и в модели OSI, данные более верхних уровней инкапсулируются в пакеты нижних уровней (см. рис. 3).

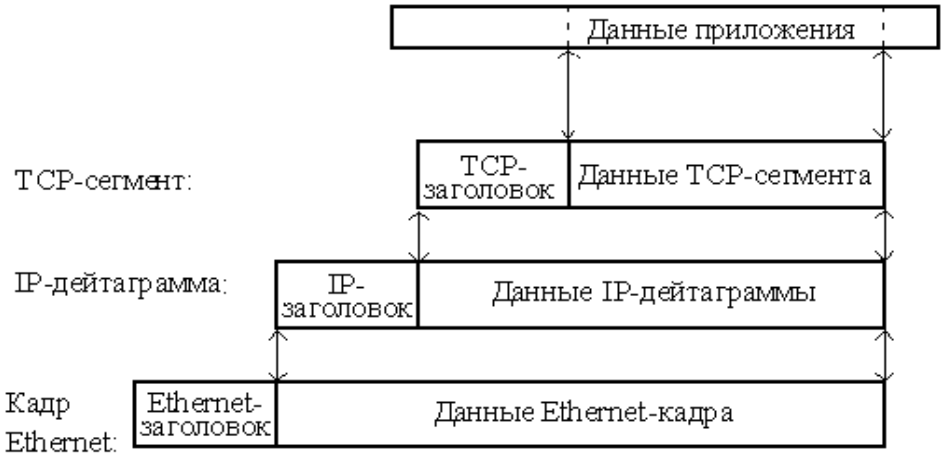

Рисунок 3 - Пример инкапсуляции пакетов в стеке

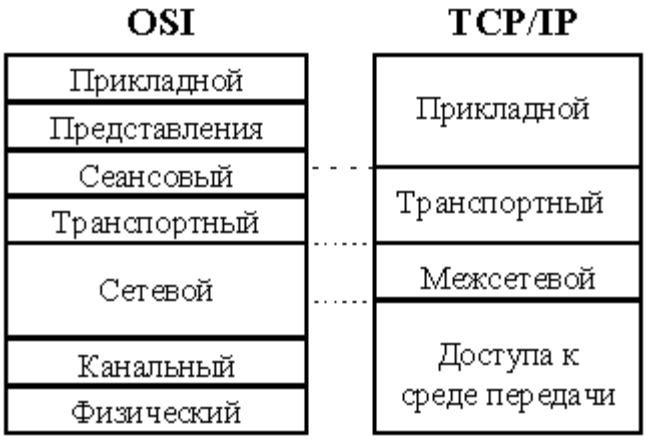

Рисунок 4 - Соотношение уровней стеков OSI и TCP/IP

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, программы работы с электронной почтой и др.

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

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* (см. далее).

Анализируя заголовок своего пакета, полученного от межсетевого уровня, транспортный модуль определяет по номеру порта получателя, какому из прикладных процессов направлены данные, и передает 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.

TCP (Transmission Control Protocol – протокол контроля передачи, *RFC 793*) – это транспортный механизм, предоставляющий поток данных, с предварительной установкой соединения, за счёт этого дающий уверенность в безошибочности получаемых данных, осуществляет повторный запрос данных в случае потери пакетов и устраняет дублирование при получении двух копий одного пакета. Естественно, что в общем случае данные не могут быть гарантировано доставлены до адресата; в таком случае клиентский процесс получает об этом уведомление.

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). Протокол TCP гарантирует, что приложение получит данные точно в такой же последовательности, в какой они были отправлены, и без потерь.

UDP (User Datagram Protocol, протокол пользовательских дейтаграмм, *RFC* 768) фактически не выполняет каких-либо особых функций дополнительно к функциям межсетевого уровня (протокола IP см. далее). Протокол UDP 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.

Межсетевой уровень и протокол IP. Основным протоколом этого уровня является протокол IP (Internet Protocol, *RFC 791*).

Протокол IP доставляет блоки данных, называемых дейтаграммами, от одного сетевого узла к другому.

В современной сети Интернет используется IP четвёртой версии, также известный как IPv4. В протоколе IP этой версии каждому узлу сети ставится в соответствие IP-адрес длиной 4 октета (иногда говорят «байта», подразумевая распространённый восьмибитовый минимальный адресуемый фрагмент памяти ЭВМ). Более подробно об IP-адресах протокола 4-й версии можно прочитать в предыдущей лабораторной работе.

В настоящее время вводится в эксплуатацию шестая версия протокола — IPv6, которая позволяет адресовать значительно большее количество узлов, чем IPv4. Эта версия отличается повышенной разрядностью адреса, встроенной возможностью шифрования и некоторыми другими особенностями. Переход с IPv4 на IPv6 связан с трудоёмкой работой операторов связи и производителей программного обеспечения и не может быть выполнен одномоментно. На начало 2007 года в Интернете присутствовало около 760 сетей, работающих по протоколу IPv6. Для сравнения, на то же время в адресном пространстве IPv4 присутствовало более 203 тысяч сетей, но в IPv6 сети гораздо более крупные, нежели в IPv4.

Данные для IP дейтаграммы передаются IP-модулю транспортным уровнем. IP-модуль предваряет эти данные заголовком, содержащим IP-адреса отправителя и получателя и другую служебную информацию, и сформированная таким образом дейтаграмма передается на уровень доступа к среде передачи (например, одному из физических интерфейсов) для отправки по каналу передачи данных.

Не все сетевые узлы могут непосредственно связаться друг с другом; часто для того, чтобы передать дейтаграмму по назначению, требуется направить ее через один или несколько промежуточных узлов по тому или иному маршруту. Задача определения маршрута для каждой дейтаграммы решается протоколом IP.

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 же 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.

Также может потребоваться, на границе сетей с различными характеристиками, разбить дейтаграмму на фрагменты, а потом собрать в единое целое на компьютере-получателе. Это 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.

Более подробную информацию об IP-адресах можно найти в предыдущей лабораторной работе.

Уровень доступа к среде передачи. Функции этого уровня:

- отображение IP-адресов в физические адреса 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 (в нашем случае это протокол IP);
- определение метода доступа к среде передачи - то есть способа, с помощью которого компьютер устанавливает свое право на произведение передачи данных (передача токена, опрос

компьютеров, множественный доступ с детектированием коллизий и т.п.).

- 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.

Обобщенная модель взаимодействия узлов на базе протоколов TCP/IP представлена на рис 5.

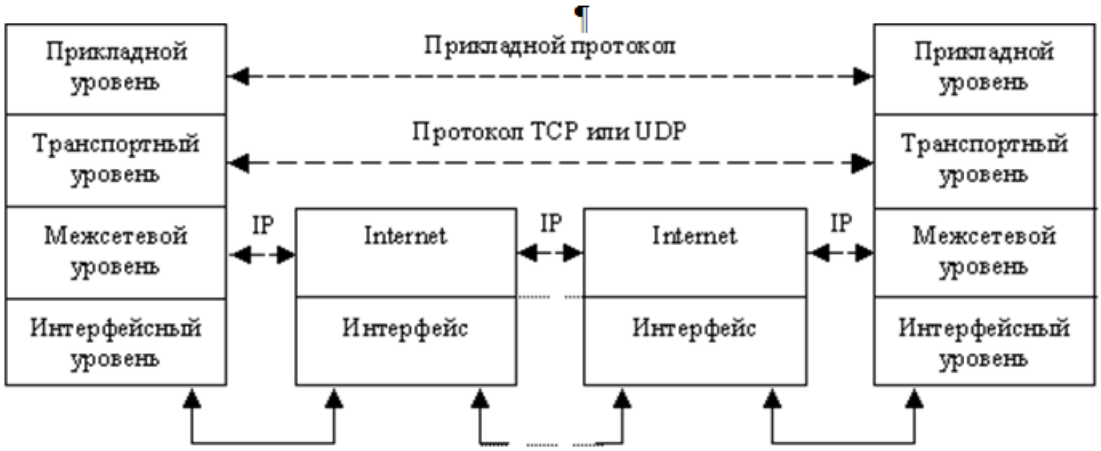

Рисунок 5 - Модель взаимодействия стеков TCP/IP

Понятие сетевых портов и сокетов

Как уже было сказано, основные прикладные сетевые сервисы используют средства транспортного уровня для взаимодействия.

Любые 2 сетевых процесса могут идентифицировать друг друга при помощи 3-х компонент: ip-адрес, протокол(TCP/UDP), порт. Часто данные компоненты носят название ***сокетами***. Сокеты – это название программного интерфейса для обеспечения информационного обмена между процессами. Т.е. для прикладных сетевых процессов взаимодействие осуществляется через сокеты. Более детально сокеты мы

будем изучать на последующих занятиях. Сейчас же рассмотрим понятие портов более подробно.

Порт – параметр протоколов TCP и UDP, определяющий пункт назначения для данных, принимаемых по сети. Порту сопоставляется номер от 1 до 65535, позволяющие различным программам, выполняемым на одном хосте, получать данные независимо друг от друга. В этом случае каждая из них обрабатывает данные, поступающие на определённый порт (иногда говорят, что программа «слушает» на том или ином порту).

Согласно IP, в каждом пакете присутствуют IP адрес узла-источника и IP адрес узла-назначения. В TCP/UDP пакетах дополнительно указываются порт источника и порт назначения. Узел назначения, получив пакет, смотрит на порт назначения и передает пакет соответствующему у себя приложению. Использование портов позволяет независимо использовать TCP/UDP протокол сразу многим приложениям на одном и том же компьютере.

Для сетевых приложений нотация указания порта следующая: «ip**:**port». Например, http://web-service.org:8888

Пояснение понятия портов представлено на рис. 6. На самом деле сетевой порт – это всего лишь числовой параметр в сетевом пакете протоколов TCP и UDP. Такие понятия как «открыть порт» означают что пакеты адресованные на данный порт будут приниматься на обработку.

Порты из диапазона 1-1024 являются привилегированными. Называются они так, потому что для их открытия (и, соответственно, запуска соответствующих сетевых сервисов) на большинстве ОС требуются права системного администратора. Большая часть привилегированных портов распределена для общеупотребительных сетевых протоколов. В табл. 1 перечислены некоторые протоколы и порты, за которыми они закреплены. Данные порты являются портами по умолчанию для соответствующих служб и чаще всего не перенастраиваются.

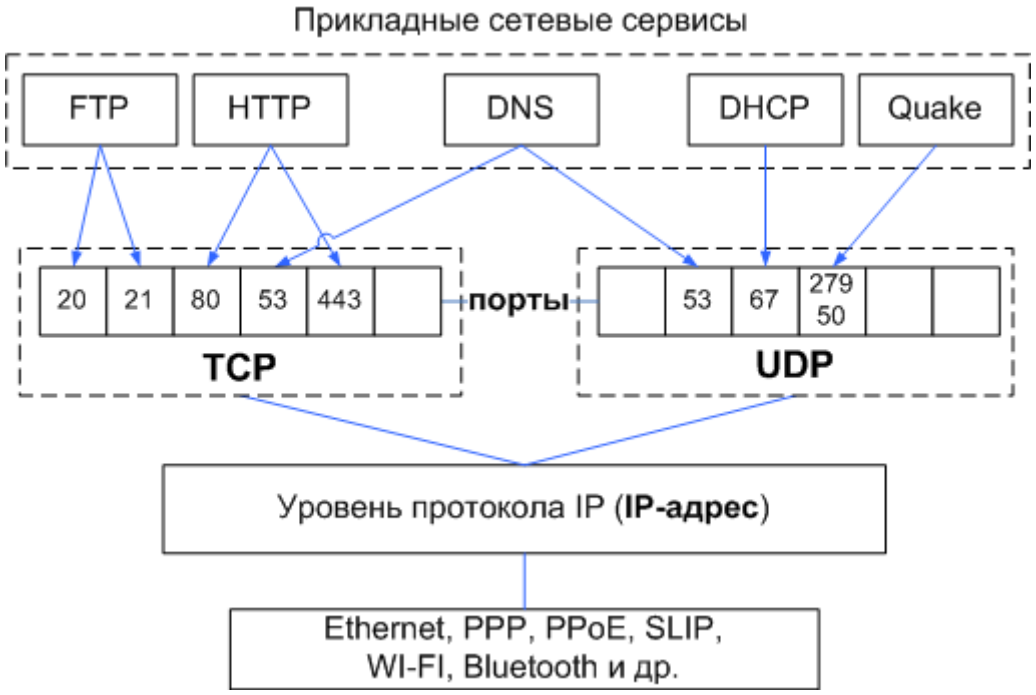

Рисунок 6. - Компоненты сокетов

Таблица 1 - Примеры некоторых стандартных сетевых портов

Порт/протокол	Сервис	Описание
20/TCP	ftp-data	Порт данных FTP
21/TCP	ftp	Порт протокола передачи файлов (File Transfer Protocol, FTP); иногда используется протоколом файловой службы (File Service Protocol, FSP)
22/TCP	ssh	Служба Безопасной ОБолочки (Secure SHell, SSH)
23/TCP	telnet	Служба Telnet
25/TCP	smtp	Протокол простой передачи почты (Simple Mail Transfer Protocol, SMTP)
53/(UDP,TCP)	domain	Службы доменных имён (такие как BIND)

80/TCP	http	Протокол передачи гипертекста (HyperText Transfer Protocol, HTTP) для служб всемирной паутины (World Wide Web, WWW)
110/TCP	pop3	Протокол почтового отделения (Post Office Protocol) версии 3
443/TCP	https	Протокол HTTP поверх SSL
992/TCP	telnets	Telnet поверх SSL (TelnetS)
993/TCP	imaps	IMAP поверх SSL (IMAPS)
994/TCP	ircs	IRC поверх SSL (IRCS)
995/TCP	pop3s	POP 3 поверх SSL (POP3S)

ВЫПОЛНЕНИЕ РАБОТЫ

Для наглядной демонстрации на практике будет рассмотрен ряд полезных сетевых утилит, которые позволяют вести мониторинг состояния стека TCP/IP.

Первой утилитой будет netstat. Данная утилита позволяет получать информацию об активных сетевых соединениях на уровне TCP и UDP протоколов, получать базовую статистику по количеству переданных и полученных пакетов уровня IP и т.д. Синтаксис команды можно почерпнуть из встроенной справки в саму утилиту («netstat /?»)

Рассмотрим несколько примеров.

«netstat -a» – получение информации обо всех установленных соединениях и открытых на прослушивание портах (см. рис. 7)

«netstat -n -b» – получение информации обо всех активных соединениях и процессах инициировавших их (см. рис. 8)

«netstat -e -s» – получение основной статистики по всем протоколам (ethernet, IPv4, IPv6, TCP, UDP) (см. рис. 9)

Рисунок 7 - Список всех установленных TCP/UDP соединений и открытых на прослушивание портов.

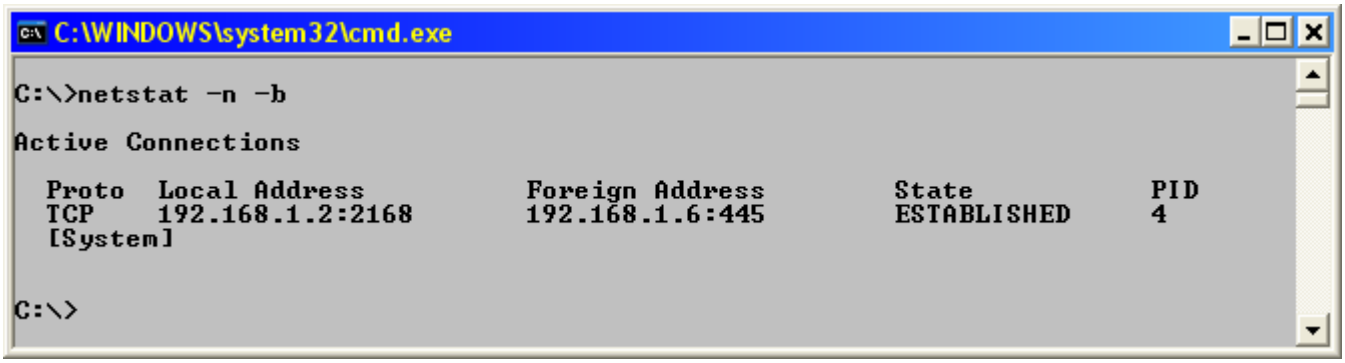

Рисунок 8 - Список активных соединений с указанием процесса

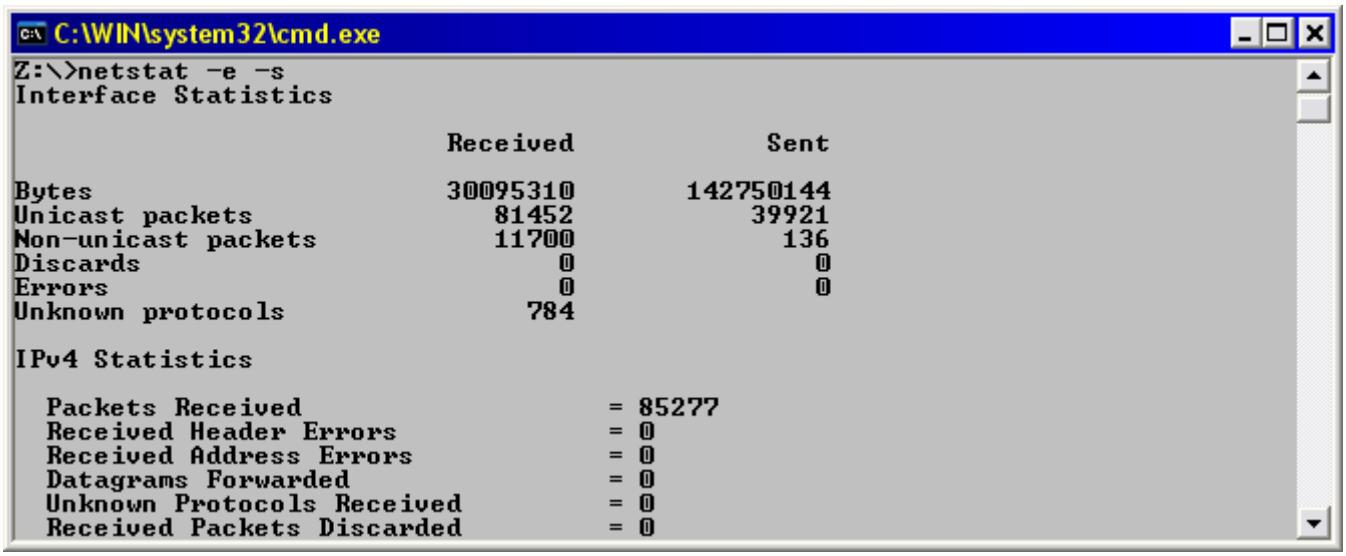

Рисунок 9 - Статистика по протоколам

Очень удобным аналогом утилиты netstat для отслеживания активных соединений является утилита TCPView (см. рис. 10). Она позволяет в интерактивном режиме отслеживать сетевые соединения, а также позволяет получать информацию о процессах, установивших соединение и завершать их в случае необходимости

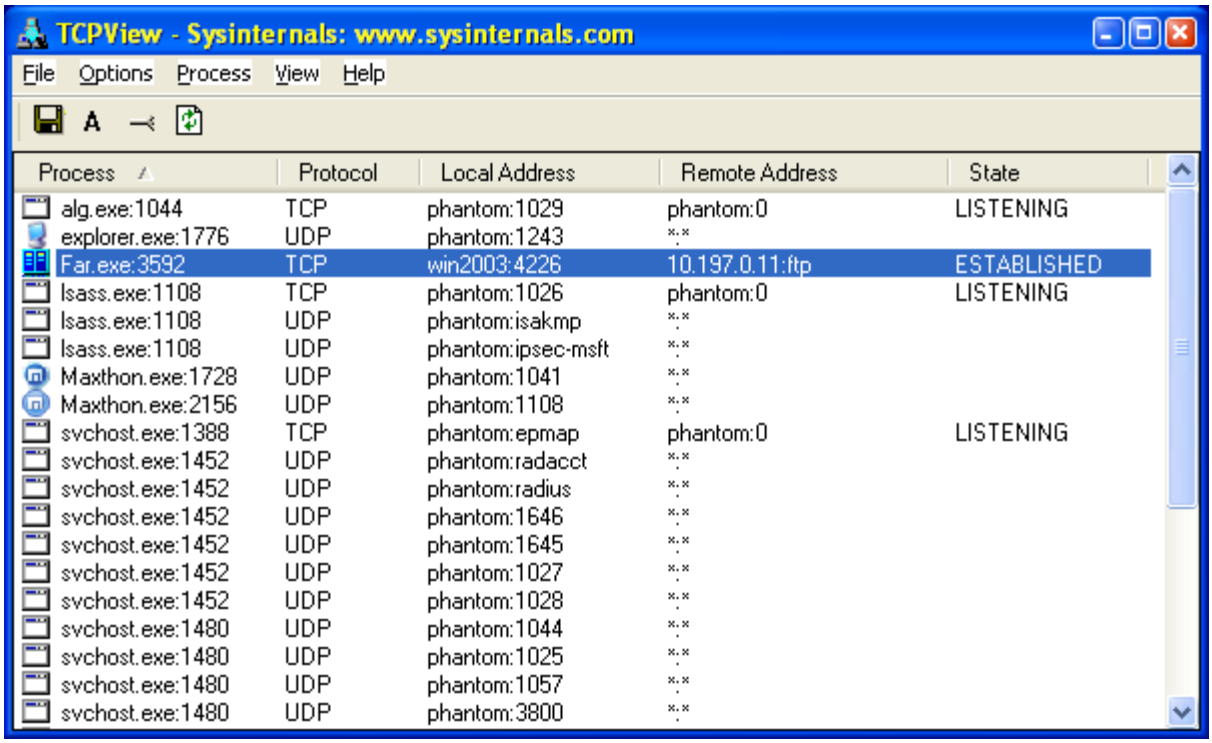

Рисунок 10 - Утилита TCPView

Часто возникает необходимость узнать, какие сетевые сервисы запущены на удаленной машине. Для решения данной задачи служат т.н. сканеры портов. Данная группа утилит позволяет с некоторой точностью узнать, какие порты открыты на удаленной машине и некоторые другие параметры. Зная номера портов зачастую можно с достаточно большой уверенностью предположить, какие сервисы запущены на удаленной машине.

Наиболее достоверная информация может быть получена для протокола TCP. Задача детектирования UDP-сервисов не всегда может быть решена. Поэтому высокая точность сканирования UDP-сервисов не гарантируется.

Одним из самых распространенных профессиональных сканеров портов является сканер «NMAP» (см. рис. 11)

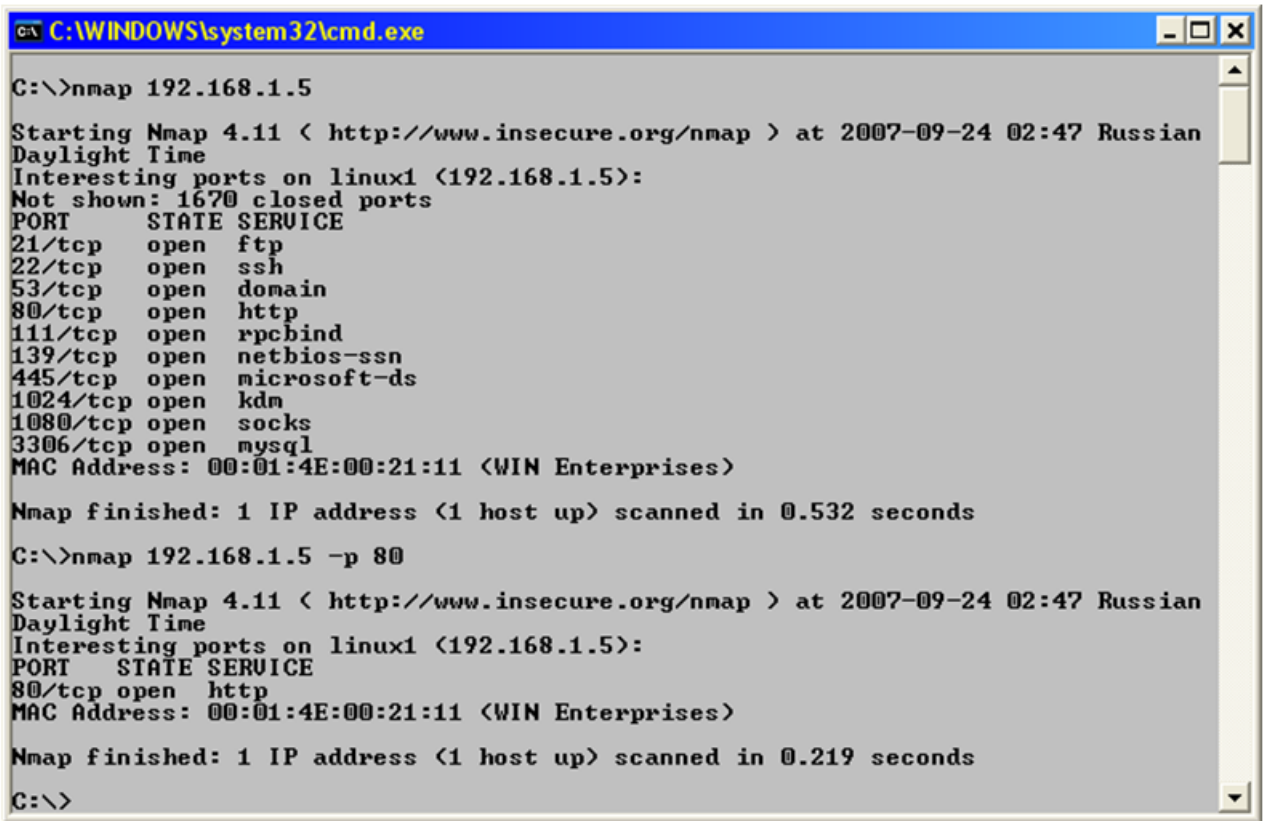

Рисунок 11 – Сканер портов NMAP

ЗАДАНИЕ

1. Изучить стек протоколов TCP/IP.
2. Найти описание протоколов IP, TCP и UDP в соответствующих RFC.
3. Изучить утилиты netstat и tcpview: проанализировать текущие сетевые соединения на сетевой машине, получить статистику по протоколам (только netstat).
4. Изучить основные команды сканера портов nmap

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Что такое сетевой протокол?
2. Зачем необходима стандартизация протоколов?
3. Понятие стека протоколов
4. Зачем введена модель OSI/ISO
5. Перечислите уровни стека протоколов TCP/IP и кратко охарактеризуйте их назначение.
6. Что такое IP-адрес?
7. В чем принципиальное отличие протоколов TCP и UDP.
8. Что такое сокет?
9. Зачем введен механизм сетевых портов?
10. Есть ли различие в протоколах реализованных, например, для ОС Windows и Linux?

СПИСОК ИНФОРМАЦИОННЫХ ИСТОЧНИКОВ

1. Танненбаум Э. Компьютерные сети. [Текст]/ Э. Танненбаум: СПб.: Питер, 2002.
2. Куроуз Дж. Ф., Росс К. В. Компьютерные сети. [Текст]/ Дж. Ф. Куроуз, К.В. Росс: СПб.: Питер, 2004.
3. TCPView for Windows [Электронный ресурс]: / Internet. - https://technet.microsoft.com/ru-ru/sysinternals/tcpview.aspx (28.09.17).