КОМПЬЮТЕРНЫЕ СЕТИ. §2

- Протоколы межсетевого уровня IP,ICMP.
- Протоколы транспортного уровня TCP, UDP.
- Протоколы прикладного уровня (на примере HTTP, FTP, SMTP).

Протоколы стека TCP\IP (1)

Протокол - набор правил, управляющий функционированием сети и обеспечивающий обмен данными. Протокол определяет:

1. Типы используемых сообщений (например, запросы и ответы)
2. Синтаксис каждого типа сообщений (поля и разделители)
3. Семантику полей (смысл информации)
4. Правила, описывающие события, которые вызывают генерацию сообщений

Уровень IV **физический + канальный**. Не регламентируется; поддерживает все популярные стандарты физического и канального уровня: Ethernet, Token Ring, FDDI, Fast Ethernet, Gigabit Ethernet, PPP, X.25, ATM и др

Уровень III **сетевой**. Межсетевое взаимодействие; доставка пакетов по составной сети, состоящей из большого количества ЛС, построенных на базе разных технологий, объединенных локальными и глобальными связями

- Базовый протокол – **IP** (Internet protocol)
- Протоколы сбора маршрутной информации **RIP** (Routing Internet Protocol) **OSPF** (Open Shortest Path First),
- Протокол межсетевых управляющих сообщений **ICMP** (Internet Control Message Protocol),
- Протокол разрешения адресов – **ARP** (Address Resolution Protocol)
- Протокол динамического конфигурирования хостов **DHCP** (Dynamic Host Configuration Protocol)

Протоколы стека TCP\IP (2)

Уровень II – **транспортный** – host-to-host-layer

- Протокол управления передачей **TCP** (Transmission Control Protocol)
- Протокол дейтаграмм пользователя **UDP** (User Datagram Protocol)

Уровень I - протоколы и сервисы прикладного уровня, обеспечивают работу сетевых приложений.

Сетевое приложение состоит из двух сторон – клиентской и серверной, которые взаимодействуют друг с другом путем обмена сообщениями.

Архитектура «клиент-сервер» описывает распределение выполнения приложения по принципу взаимодействия двух программных процессов, один из которых в этой модели называется «клиентом», а другой – «сервером». Клиентский процесс запрашивает некоторые услуги, а серверный процесс обеспечивает их выполнение. При этом один серверный процесс может обслуживать множество клиентских процессов.

SMTP, HTTP, FTP регламентируют обмен данными между клиентом и сервером

DNS – доменная служба имен (Domain name service)

Протоколы сетевого уровня. IPv4 (1)

IP – Internet Protocol

- поддержание интерфейса (взаимодействия) с технологиями сетей, образующих составную сеть (нижележащий уровень)

- поддержание интерфейса с транспортными протоколами.

В каждой очередной сети, лежащей на пути следования пакета, протокол IP обращается к средствам транспортировки этой сети, чтобы с их помощью передать пакет на маршрутизатор, ведущий к следующей сети либо на узел получателя.

Протокол без установления соединения. Реализует доставку пакета «по возможности» (с максимальными усилиями). Не предусмотрены повторные передачи при сбоях, нет механизмов обеспечения достоверности информации.

Поддерживает обработку каждого IP-пакета как независимой единицы обмена, не связанной с другими пакетами.

Формат заголовка пакета IP:

Version, 4 HLEN, 4 Type of Service, 8 Total Length, 16
Identification, 16 Flags, 3 Fragment Offset, 13
Time to Live, 8 HL Protocol, 8 Header Checksum, 16
Source Address, 32
Destination Address, 32
Options Padding
Data

Протоколы сетевого уровня. IPv4 (2)

Version, 4 HLEN, 4 Type of Service, 8 Total Length,16
PRI(3);D;T;R;res/2bit
Identification, 16 Flags, 3 Fragment Offset, 13
Time to Live, 8 HL Protocol, 8 Header Checksum, 16

Номер версии (VERS) указывает версию протокола IP. Сейчас 4, будет 6.

Длина заголовка (HLEN) длина заголовка в 32-битовых словах. Обычно заголовок имеет длину в 20 байт (пять 32-битовых слов), но может быть больше, если используется поле Резерв (IP OPTIONS).

Тип сервиса (SERVICE TYPE): приоритетность пакета + выбор маршрута.

Первые три бита образуют подполе приоритета пакета (PRECEDENCE). Приоритет может иметь значения от 0 (нормальный пакет) до 7 (пакет управляющей информации). **DS-service – современное название**

Тип сервиса - три бита, определяющие критерий выбора маршрута.

- Установленный бит **D** (Delay) - маршрут должен выбираться для минимизации задержки доставки данного пакета,
- бит **T** (Throughout) - для максимизации пропускной способности,
- бит **R** (Reliability) - для максимизации надежности доставки.
- два бита резервные

Общая длина (TOTAL LENGTH) 2 байта - общая длина пакета с учетом заголовка и поля данных

Протоколы сетевого уровня. IPv4 (3)

Version, 4 HLEN, 4 Type of Service, 8 Total Length,16
PRI(3);D;T;R;res/2bit
Identification, 16 Flags, 3 Fragment Offset, 13
[res/1bit;D;M]
Time to Live, 8 HL Protocol, 8 Header Checksum, 16

Идентификатор пакета (IDENTIFICATION) – распознавание пакетов, образовавшихся путем фрагментации исходного пакета. Все фрагменты должны иметь одинаковое значение этого поля.

Флаги (FLAGS) указывает на возможность фрагментации пакета.

- Установленный бит **Do not Fragment** (**DF**) запрещает маршрутизатору фрагментировать данный пакет;
- установленный бит **More Fragments** (**MF**) указывает, что это промежуточный фрагмент.

Смещение фрагмента (FRAGMENT OFFSET) – указывает в байтах величину **смещения поля данных пакета** от начала общего поля данных исходного пакета, подвергнутого фрагментации. Используется при сборке/разборке фрагментов пакетов при передачах их между сетями с различными величинами максимальной длины пакета.

Протоколы сетевого уровня. IPv4 (4)

Identification, 16 **Flags, 3** **Fragment Offset, 13**
Time to Live, 8 **HL Protocol, 8** **Header Checksum, 16**
Source IP Address, 32
Destination IP Address, 32
Options **Padding**

Время жизни (TIME TO LIVE) – **предельный срок (в сек.)**, в течение которого пакет может перемещаться по сети. Время жизни данного пакета задается источником передачи средствами протокола IP. На шлюзах и в других узлах каждую секунду из текущего времени жизни вычитается 1; единица вычитается также при каждой транзитной передаче (даже если не прошла секунда). Далее пакет аннулируется.

Идентификатор Протокола верхнего уровня (PROTOCOL) указывает, какому протоколу верхнего уровня принадлежит пакет (TCP, UDP и т.п.)

Контрольная сумма (HEADER CHECKSUM) всего заголовка, 2 байта.

Адрес источника (SOURCE IP ADDRESS) 32 бита

Адрес назначения (DESTINATION IP ADDRESS) 32 бита

Резерв (IP OPTIONS) необязателен; используется обычно при отладке сети. Число подполей может быть произвольным.

Выравнивание (PADDING) добавляется в конце поля Резерв для выравнивания заголовка пакета по 32-битной границе. Переменная длина.

Протоколы сетевого уровня. ICMP

Протокол ICMP - internet control message protocol,

Протокол **ICMP** передачи команд и сообщений об ошибках выполняет диагностические функции, используется программным обеспечением ЭВМ при взаимодействии элементов сети друг с другом в рамках идеологии TCP/IP.

ICMP-протокол сообщает об ошибках в IP-дейтограммах, но не дает информации об ошибках в самих ICMP-сообщениях.

ICMP-протокол осуществляет:

- контроль времени жизни дейтограмм в системе;
- реализует переадресацию пакета;
- выдает сообщения о недостижимости адресата или о некорректности параметров;
- формирует и пересылает временные метки;
- выдает запросы и отклики для служебной информации разного рода

ICMP-сообщения об ошибках никогда не выдаются в ответ на:

- ICMP-сообщение об ошибке.
- При широковещательной адресации.
- Для фрагмента дейтограммы (кроме первого).
- Для дейтограмм, чей адрес отправителя является нулевым, широковещательным.

Протоколы транспортного уровня. Порты

Протоколы транспортного уровня

- TCP - Протокол управления передачей
- UDP - Протокол пользовательских дейтаграмм

Оба протокола поддерживают интерфейс

- **с вышележащим прикладным уровнем**, передавая данные на входной интерфейс хоста соответствующему приложению
- **с нижележащим сетевым уровнем**, упаковывая свои PDU в IP-пакеты.

Каждый компьютер может выполнять несколько процессов, поэтому, когда IP-пакет средствами протокола IP доставлен на сетевой интерфейс компьютера-получателя – данные должны быть переправлены конкретному процессу.

Процедура приема протоколами TCP\UDP данных, поступающих на отправку от нескольких различных прикладных служб, называется мультиплексированием. Обратная процедура (распределение поступивших с сетевого уровня данных по прикладным процессам) называется демультиплексированием.

Для связи с прикладными процессами используются **порты**. Протоколы TCP\UDP для каждого прикладного процесса ведут две очереди для поступающих и отправляемых данных. Совокупность входной и выходной очередей для каждого процесса называется портом.

Номера портов используются для идентификации приложений (независимая нумерация TCP-портов и UDP-портов).

Протоколы транспортного уровня. TCP (1)

Номера некоторых портов зарезервированы (TCP: 21 – FTP, 22 – SSH, 23 – telnet, 25 – smtp, 80 – http; UDP: 53 - DNS), или используются произвольные (0-1023).

Пара «IP-адрес – TCP\UDP-порт» однозначно определяет прикладной процесс в сети и называется TCP\UDP-сокет (socket).

Протокол TCP *(Transmission Control Protocol)*

В стеке TCP/IP протокол TCP обеспечивает надежную транспортировку данных между прикладными процессами путем установления логического соединения.

Единицей данных протокола TCP является сегмент.

Информация, поступающая к протоколу TCP в рамках логического соединения от протоколов более высокого уровня, рассматривается как неструктурированный поток байтов.

Поступающий поток буферизуется средствами TCP и делится на сегменты: для передачи на сетевой (IP) уровень из буфера "вырезается" некоторая непрерывная часть данных размером 2460 байт, называемая сегментом.

Протоколы транспортного уровня. TCP (2)

Для организации надежной передачи данных устанавливается *логическое соединение* между двумя прикладными процессами.

В рамках соединения осуществляется обязательное подтверждение правильности приема для всех переданных сообщений, при необходимости выполняется повторная передача.

Соединение в TCP позволяет вести передачу данных одновременно в обе стороны, то есть **полнодуплексную передачу**.

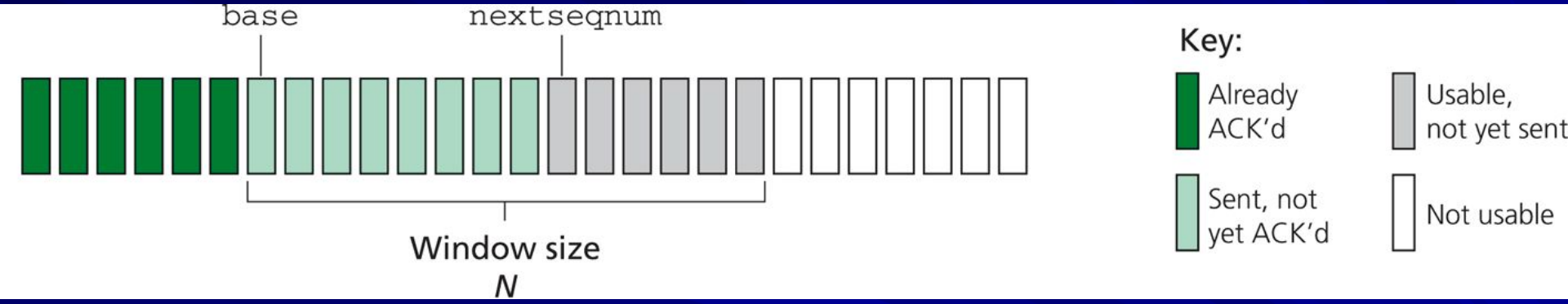
- Сторона-инициатор посылает запрос к TCP на открытие порта для передачи.
- После открытия порта протокол TCP на стороне процесса-инициатора посылает запрос процессу, с которым требуется установить соединение.
- Протокол TCP на приемной стороне открывает порт для приема данных и возвращает квитанцию, подтверждающую прием запроса.
- Протокол на приемной стороне открывает порт для передачи и также передает запрос к противоположной стороне.
- Сторона-инициатор открывает порт для приема и возвращает квитанцию.
- Соединение считается установленным. Происходит обмен данными.

В рамках соединения правильность передачи каждого сегмента должна подтверждаться **квитанцией получателя**. Квитирование – один из традиционных методов обеспечения надежной связи.

Протоколы транспортного уровня. TCP (3)

Количество кадров, которые разрешается передавать без получения на эти кадры ответных квитанций, называется **размером окна**.

В протоколе TCP реализован алгоритм квитирования с использованием **скользящего окна**. Окно определено на множестве нумерованных байт неструктурированного потока данных, поступающих с верхнего уровня и буферизуемых протоколом TCP.



В заголовок каждого сегмента помещается номер байта в потоке передаваемых данных, разбитых на сегменты – последовательный номер.

На основании этого номера TCP-получатель

(а) отличает сегмент от других,

(б) позиционирует его относительно общего потока,

(в) может сделать вывод, что сегмент является дубликатом или что между двумя сегментами пропущены данные.

Протоколы транспортного уровня. TCP (4)

В качестве квитанции посылается сегмент, содержащий номер последнего байта в принятом сегменте +1, что совпадает с номером первого байта следующего сегмента.

Квитанция посылается только в случае правильного приема данных, отрицательные квитанции не посылаются. Таким образом, отсутствие квитанции означает либо прием искаженного сегмента, либо потерю сегмента, либо потерю квитанции.

При установлении соединения и в ходе приема\передачи обе стороны, выступая в роли получателя, посылают друг другу ***окна приема*.**

Если сегмент выходит за рамки окна, передачу необходимо приостановить до прихода следующей квитанции.

В одном сегменте могут быть помещены и данные, и квитанция, подтверждающая прием от той стороны, куда идут данные.

Получатель может отправить квитанцию, подтверждающую прием сразу нескольких сегментов, если они образуют непрерывный поток байтов.

Копия переданного сегмента помещается в очередь повторной передачи. При получении квитанции копия удаляется. Если квитанция не приходит в течение заданного времени – сегмент посылается повторно. Если повторный сегмент пришел, когда исходный уже на месте – дубликат отбрасывается.

Если пропущен сегмент – посылается повторно квитанция на последний полученный пакет.

Протоколы транспортного уровня. TCP (5)

Выбор времени ожидания (тайм-аут) очередной квитанции влияет на производительность протокола TCP.

- Тайм-аут не должен быть слишком коротким, чтобы исключить избыточные повторные передачи, которые снижают полезную пропускную способность системы.
- Но он не должен быть и слишком большим, чтобы избежать длительных простоев, связанных с ожиданием квитанции.
- Специальные алгоритмы динамического регулирования размера окна для контроля перегрузки.

Алгоритм выбора тайм-аута, принятый в TCP.

- При каждой передаче засекается **время оборота** – интервал от момента отправки сегмента до получения квитанции о приеме.
- Получаемые значения усредняются с весовыми коэффициентами, возрастающими от предыдущих замеров к последующим, чтобы усилить влияние последних измерений.
- В качестве тайм-аута выбирается усредненное время оборота, умноженное на некий эмпирический коэффициент (<2).
- В сетях с большим разбросом времени оборота необходимо учитывать также дисперсию.

Протоколы транспортного уровня. TCP (6)

Итак, варьируя величину окна, можно влиять на загрузку сети:

Чем больше окно, тем большую порцию неподтвержденных данных можно послать в сеть. Если сеть не справляется с нагрузкой, то возникают очереди в промежуточных и конечных узлах.

С другой стороны, слишком малый размер окна ограничивает скорость передачи данных, которая определяется временем путешествия в сети каждого сообщения.

Существуют разные схемы регулирования размера окна как со стороны получателя, так и со стороны отправителя.

Например, если отправитель фиксирует ненадежную связь (например, долго идут квитанции), он может уменьшить окно по собственной инициативе.

Формат сообщений TCP. Сегменты протокола TCP состоят из заголовка и блока данных. Формат заголовка следующий:

Source Port, 2 байта **Destination Port, 2 байта**

Sequence Number, 4 байта

Acknowledgment Number, 4 байта

Hlen (4бит) Reserved (6бит) Code bits (6бит) Window 2 байта

Checksum, 2 байта **Urgent Pointer, 2байта**

Options, max 3 байта **Padding**

Протоколы транспортного уровня. TCP (7)

Source Port, 2 байта **Destination Port, 2 байта**
Sequence Number, 4 байта
Acknowledgment Number, 4 байта
Hlen, 4бит **Reserved, 6бит** **Code bits (6bits)** **Window, 2 байта**
[URG ACK PSH RST SYN FIN]

Порт источника (Source Port) идентифицирует процесс-отправитель;
Порт назначения (Destintion Port) идентифицирует процесс-получатель;
Последовательный номер (Sequence Number) - номер байта, определяющего смещение сегмента относительно начала потока отправляемых данных;
Подтвержденный номер (Acknowledgment Number) - максимальный номер байта в полученном сегменте +1; используется как квитанция;
Длина заголовка (HLEN) – длина заголовка сегмента – количество 32-битовых слов;
Резерв (RESERVED) поле зарезервировано для будущего использования;
Кодовые биты (CODE BITS) - служебная информация о типе сегмента

- URG - срочное сообщение;
- ACK - квитанция на принятый сегмент;
- PSH - запрос на отправку сообщения без ожидания заполнения буфера;

Протоколы транспортного уровня. TCP (8)

Source Port, 2 байта **Destination Port, 2 байта**
Sequence Number, 4 байта
Acknowledgment Number, 4 байта
Hlen, 4бит **Reserved, 6бит** **Code bits (6bits)** **Window, 2 байта**
[URG ACK PSH RST SYN FIN]
Checksum, 2 байта **Urgent Pointer, 2байта**
Options, max 3 байта **Padding**

- RST - запрос на восстановление соединения;
- SYN – синхронизация счетчиков переданных данных при установлении соединения;
- FIN - признак достижения передающей стороной последнего байта в потоке передаваемых данных.

Указатель срочности (URGENT POINTER) используется совместно с кодовым битом URG, указывает на конец данных, которые необходимо срочно принять, несмотря на переполнение буфера;

Опции (OPTIONS) - поле имеет переменную длину, может отсутствовать, максимальная величина 3 байта; используется для решения вспомогательных задач, например, при выборе максимального размера сегмента;

Заполнитель (PADDING) переменная длина, фиктивное поле для доведения размера заголовка до целого числа 32-битовых слов.

Протоколы транспортного уровня. UDP

Протокол UDP реализует сервис по возможности, не гарантирует доставку своих сообщений, т.е. не компенсирует ненадежность протокола IP.

- Отсутствие процедуры логического соединения
- Отсутствие информации о состоянии соединения
- Небольшой размер заголовка
- Данные снабжаются заголовком и сразу передаются на IP уровень
- Не располагает средствами контроля перегрузки; не может приостановить передачу данных, если между получателем и отправителем имеются перегруженные линии связи.
- годится для групповой и широковещательной рассылки

Формат сообщений UDP

Source Port Destination Port

Hlen Checksum

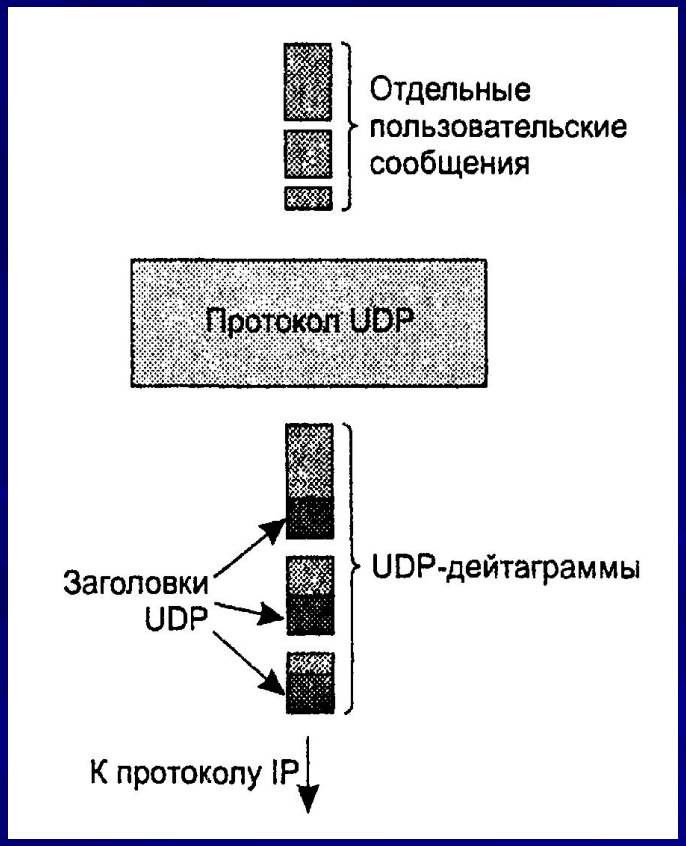
Data

Приложения на базе UDP

- TFTP – протокол передачи данных на базе UDP
- NFS – удаленный файловый сервер
- Протоколы пересылки потоковых мультимедиа (Real Networks и др)
- Протоколы Интернет-телефонии (например, DialPad)
- Служба DNS

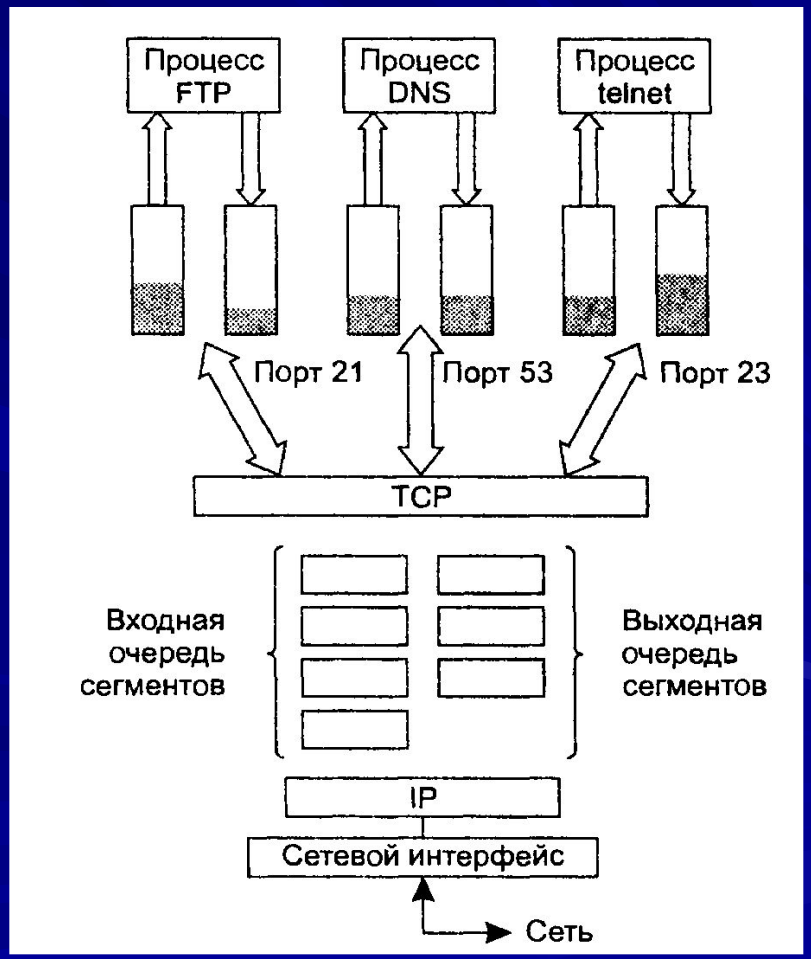
Формирование UDP-дейтаграммы

Каждая UDP-дейтаграмма переносит отдельное пользовательское сообщение, т.е. ее длина не может превышать поля данных протокола IP, которое в свою очередь ограничено размером кадра технологии нижнего уровня.



Формирование TCP-сегментов

Поступающий неструктурированный поток байт буферизуется средствами TCP и делится на сегменты по 2460 байт, каждый из которых снабжается TCP-заголовком



Протоколы прикладного уровня (1)

Протокол - набор правил, управляющий функционированием сети и обеспечивающий обмен данными

Протокол определяет:

1. Типы используемых сообщений (например, запросы и ответы)
2. Синтаксис каждого типа сообщений (поля и разделители)
3. Семантику полей (смысл информации)
4. Правила, описывающие события, которые вызывают генерацию сообщений

Сетевое приложение состоит из двух сторон – клиентской и серверной, которые взаимодействуют друг с другом путем обмена сообщениями.

Архитектура «клиент-сервер» описывает распределение выполнения приложения по принципу взаимодействия двух программных процессов, один из которых в этой модели называется «клиентом», а другой – «сервером». Клиентский процесс запрашивает некоторые услуги, а серверный процесс обеспечивает их выполнение. При этом один серверный процесс может обслуживать множество клиентских процессов.

Протоколы SMTP, HTTP, FTP регламентируют обмен данными между клиентом и сервером

Протоколы прикладного уровня (2)

HTTP (Hyper Text Transfer Protocol) – передача объектов между веб-клиентом и веб-сервером

Браузер – агент пользователя HTTP (Internet explorer et al.) – клиентская сторона протокола. Веб-сервер – серверная сторона.

Веб-страница включает базовый html-файл и объекты: картинки, приложения и др., обладающие url-адресами (uniform resource locator).

Протокол определяет, каким образом клиенты запрашивают веб-страницы и каким образом веб-серверы осуществляют передачу этих страниц.

Используется TCP как протокол транспортного уровня.

Непостоянное соединение. После каждого запроса+ответа TCP-соединение закрывается. Параллелизм – браузер устанавливает максимально возможное число параллельных соединений (можно установить степень параллелизма 1, стандартно 5-10). Недостатки – каждый раз требуется время время на обмен предварительными сообщениями.

Постоянные соединения. Закрытие соединения происходит, если оно не используется в течение установленного интервала времени – интервала ожидания.

Конвейеризация – запрос формируется сразу после обнаружения ссылки на объект и направляется к серверу, не дожидаясь выполнения предыдущих запросов.

Протоколы прикладного уровня (3)

FTP – File Transfer Protocol.

Пользователь взаимодействует с ftp-агентом (клиентом) на своем хосте: указывает имя удаленного хоста, вводит имя и пароль, которые передаются серверу с помощью ftp-команд. Далее, начинается процесс передачи файлов.

Используются два параллельных соединения –

(1) управляющее для пересылки управляющими сообщениями

(2) для собственно обмена данными.

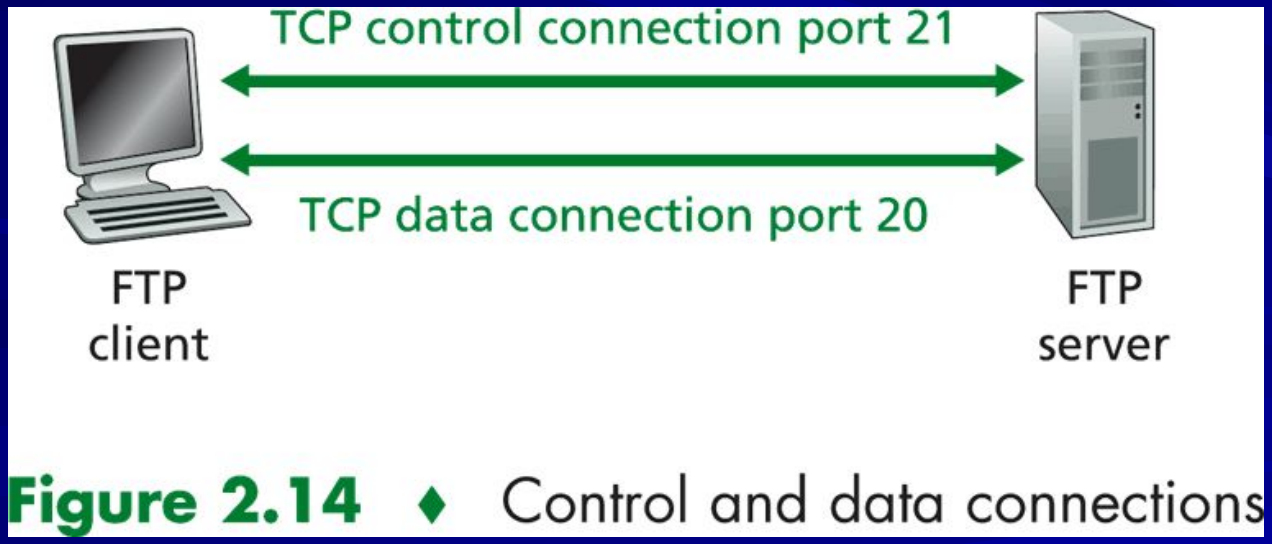


Figure 2.14 ♦ Control and data connections

Протоколы прикладного уровня (4)

SMTP – Simple Mail Transfer Protocol – главный протокол прикладного уровня для электронной почты.

Используется механизм надежной передачи протокола TCP. Как и большинство протоколов прикладного уровня – протокол двухсторонний, регламентирует взаимодействие двух почтовых серверов.

Сторона клиента выполняется на почтовом сервере отправителя, сторона сервера – на почтовом сервере получателя.

Структура электронной почты включает три основных компоненты:

(1) агенты пользователя: «почтовые программы»

(2) почтовые серверы

(3) протокол SMTP, регламентирует взаимодействие почтовых серверов.

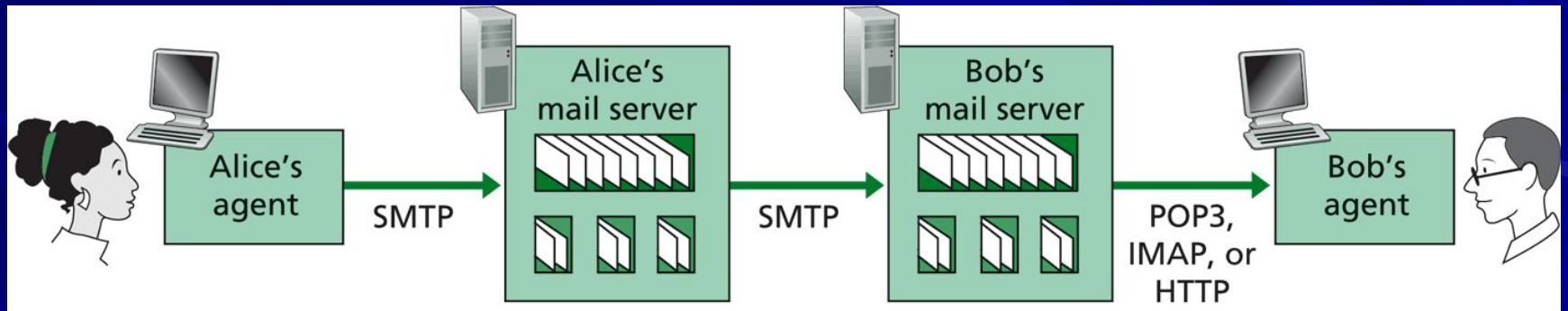


Figure 2.17 ♦ E-mail protocols and their communicating entities

Протоколы прикладного уровня (5)

С помощью агента отправителя создается письмо и направляется почтовому серверу, где письмо попадает в очередь исходящих сообщений.

Почтовый сервер отправителя посылает письмо почтовому серверу получателя, где оно помещается в его почтовый ящик (SMTP).

Получатель с помощью своей почтовой программы подключается к своему почтовому серверу и получает письмо.

При сбое отправки почтовый сервер отправителя сохраняет письмо в очереди исходящих сообщений и повторяет попытки через определенные интервалы времени (SMTP).

После заданного числа неудачных попыток письмо уничтожается, отправитель получает соответствующее уведомление (SMTP).

В отличие от HTTP, требует простой ASCII кодировки символов как в заголовке, так и в теле сообщения (осталось с 80х). Это приводит к необходимости бинарных файлов с графической и мультимедийной информацией в эту кодировку, а затем обратного преобразования в бинарную форму.

SMTP не предусматривает передачу сообщений через промежуточные почтовые серверы.

Протоколы доступа к электронной почте регламентируют взаимодействие почтового сервера с хостом.

IMAP – Internet Mail Access Protocol (отправка на сервер);

POP3 – Post Office Protocol v3 (доставка на хост), **HTTP**.