Введение "Общие сведения о стеке протоколов TCP/IP"

(начало)

1. Краткая справка по стеку протоколов TCP/IP

Несмотря на отсутствие универсальных правил описания TCP/IP посредством многоуровневой модели, обычно в стеке TCP/IP верхние 3 уровня (прикладной, представительный и сеансовый) модели OSI объединяют в один — прикладной. Поскольку в таком стеке не предусматривается унифицированный протокол передачи данных, функции по определению типа данных передаются приложению. Интерпретация стека TCP/IP обычно содержит от 3 до 5 уровней, но наиболее часто в него включают 4 уровня.

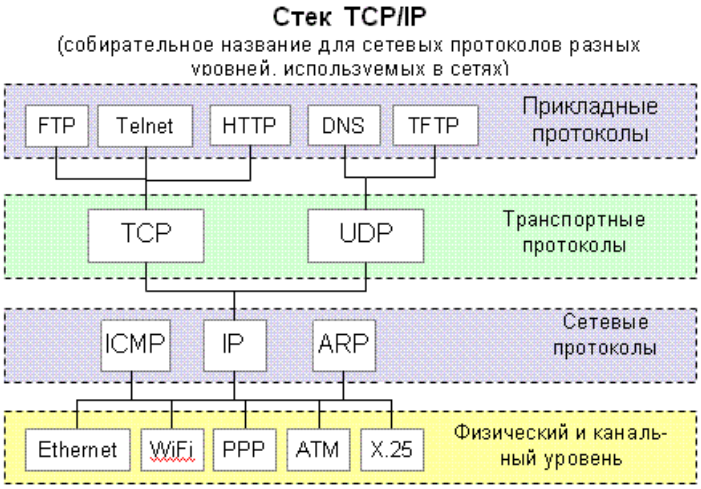


Рис. 1. Архитектура семейства протоколов TCP/IP.

Прикладные протоколы работают на верхнем уровне модели OSI и обеспечивают взаимодействие приложений и обмен данными между ними.

Транспортные протоколы поддерживают сеансы связи между компьютерами и гарантируют надежный обмен данными между ними.

Сетевые протоколы обеспечивают услуги связи. Эти протоколы управляют: адресацией, маршрутизацией, проверкой ошибок и запросами на повторную передачу

Передача данных по сети

Данные передаются вниз по стеку при отправке в сеть и вверх по стеку при получении из сети. Четырехуровневая структура TCP/IP проявляется в способе обработки данных при их прохождении вниз по стеку, от прикладного уровня непосредственно к физической сети. Каждый уровень стека добавляет управляющую информацию, гарантируя корректную доставку. Блок управляющей информации называется заголовком (header), поскольку предшествует передаваемым данным. Каждый уровень интерпретирует всю информацию, полученную от вышележащего уровня, в качестве данных и добавляет к этим данным собственный заголовок. Дополнение информации по доставке на каждом уровне носит название инкапсуляции.

При получении данных происходит обратный процесс. Каждый уровень удаляет соответствующий заголовок и передаёт данные вышележащему уровню. При передаче вверх по стеку информация, получаемая от

нижележащих уровней, интерпретируется в качестве заголовка и сопутствующих данных (рис. 2).

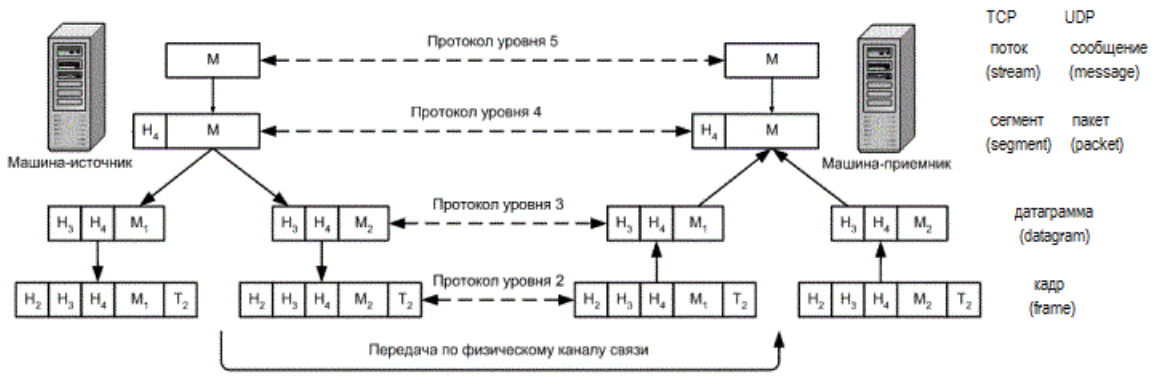


Рис. 2. Принцип иерархического построения протоколов.

Каждому уровню соответствуют определенные структуры данных. Теоретически уровень не обязан знать о структурах данных, применяемых на соседних уровнях, однако на практике структуры данных уровня проектируются таким образом, чтобы хорошо сочетаться со структурами «соседей», в целях повышения эффективности передачи данных. Тем не менее, каждому уровню соответствует собственная структура данных и специальная терминология её описания.

Основные термины для обозначения передаваемых данных

- На прикладном уровне приложения TCP считают данные потоком (stream), а приложения UPD – сообщением (message).
- На транспортном уровне данные TCP хранятся в сегментах (segment), а данные UPD – в пакетах (packet).
- Межетевой уровень рассматривает данные в качестве блоков, называемых датаграммами (datagrams).

Многочисленные типы сетей, поверх которых работает TCP/IP, также используют разнообразную терминологию в области передаваемых данных. В большинстве сетей приняты термины пакет или фрейм. Этой терминологии придерживается и пользовательский интерфейс анализатора Wireshark. Перехваченный на уровне сетевого интерфейса (уровне доступа к сети) блок данных именуется пакетом или фреймом.

Основные протоколы стека TCP/IP

- ARP – Отвечает за получение MAC адреса хоста, размещённого в текущей сети, по его IP адресу. Использует broadcast.
- ICMP – Посылка сообщений об ошибках, обнаруженных в процессе передачи пакетов.
- IGMP – Информирует маршрутизаторы о наличии в данной сети multicast группы.
- IP – Обеспечивает передачу и маршрутизацию пакетов.
- TCP – Обеспечивает соединение между двумя хостами, с гарантируемой доставкой данных, разбитых на последовательность фрагментов.
- UDP – Протокол пользовательских датаграмм для передачи данных в сетях IP без установления соединения. Один из самых простых протоколов транспортного уровня. В отличие от TCP, не гарантирует доставку пакета, но позволяет гораздо быстрее и эффективнее доставлять данные для приложений, которым требуется большая пропускная способность линий связи, либо требуется малое время доставки данных.

2. Уровень доступа к сети

Канальный уровень (англ. data link layer) предназначен для обеспечения взаимодействия сетей по физическому уровню и контролем над ошибками, которые могут возникнуть. Полученные с физического уровня данные, представленные в битах, он упаковывает в кадры, проверяет их на целостность и, если нужно, исправляет ошибки (формирует повторный запрос поврежденного кадра) 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 два подуровня: MAC (англ. media access control) регулирует доступ к разделяемой физической среде, LLC (англ. logical link control) обеспечивает обслуживание сетевого уровня. На этом уровне работают коммутаторы, мосты и другие устройства. В программировании этот уровень представляет драйвер сетевой платы.

Функциональность данного уровня включает инкапсуляцию IP-дейтаграмм в передаваемые по сети фреймы, а также отображение IP-адресов в физические адреса, применяемые в сети.

Для передачи данных по сети хост должен знать MAC адрес хоста, которому передаются данные. Для получения MAC адреса по известному IP адресу служит протокол ARP (англ. Address Resolution Protocol — протокол определения адреса).

Разрешение локального IP адреса

- ARP запрос возникает всегда, когда хост пытается связаться с другим хостом. Если IP определит, что IP адрес получателя находится в локальной сети, то сначала хост проверяет наличие этого адреса в собственном кэше ARP.
- В случае неудачи ARP строит запрос и посылает его широковещательным сообщением всем хостам подсети. Запрос имеет следующую структуру: “Хост с IP адресом a.b.c.d, сообщите мне Ваш MAC адрес”. Запрос также содержит информацию об IP и MAC адресе отправителя.
- Каждый хост в подсети получает запрос и проверяет на соответствие свой IP адрес. Если он не совпадает с указанным в запросе, то запрос игнорируется.
- Если IP адрес, указанный в запросе, совпадает с IP адресом хоста, то хост посылает ARP ответ непосредственно отправителю, используя его MAC адрес. И заносит информацию об IP/MAC адресе отправителя в ARP кэш.

Разрешение удаленного IP адреса

В этом случае инициируется ARP запрос маршрутизатору, который пересылает датаграммы в сеть назначения.

- Если IP определит, что IP адрес получателя не находится в локальной сети, то сначала проверяется наличие этого адреса в таблице маршрутизации. В случае неудачи происходит попытка найти MAC адрес default gateway в ARP кэше.
- Если MAC адрес default gateway не найден, то формируется ARP запрос на определение его MAC адреса. После получения ответа хост посылает информацию через default gateway в сеть назначения.
- Когда запрос приходит в сеть назначения, то маршрутизатор определяет MAC адрес получателя (см. разрешение локального IP адреса) и посылает

ICMP ответ маршрутизатору хоста отправителя.

(начало)

3. Межсетевой уровень

Наиболее важный протокол этого уровня – протокол Internet (IP), Он обеспечивает работу базовой службы доставки пакетов, на которой построены сети TCP/IP. Все протоколы этого и соседствующих (верхних) уровней используют протокол IP для доставки данных. Все входящие и исходящие потоки данных проходят через IP независимо от пункта назначения.

Протокол управляющих сообщений (ICMP - Internet Control Message Protocol) использует функциональность доставки дейтаграмм для отправки собственных сообщений. Сообщения ICMP выполняют следующие информативные, управляющие и связанные с ошибками функции TCP/IP:

- Управление потоками данных (Flow control);
- Обнаружение недостижимых адресатов;
- Перенаправление маршрутов;
- Проверка состояния удалённых узлов.

3.1. Сетевой протокол IP

Стандартная модель функционирования протокола IP достаточно проста. Он принимает данные с транспортного уровня модели OSI, разбивает их на небольшие пакеты данных, которые передает на канальный уровень. На приемной стороне буферирует пакеты для дальнейшей их сборки в исходный вид.

Для организации гарантированной доставки каждого IP-пакета в их заголовки помещаются адреса отправителя и получателя, а также вычисляется значение контрольной суммы. Структура заголовка пакета приведена на рис. 3, а ниже дано описание его полей.

Байты	0		1	2-3	4-5	6	6-7	8	9	10-11	12-15	16-19	20	
Биты	0-3	4-7	0-7	0-15	0-15	0-2	3-15	0 -7	0 -7	0 -15	0 -31	0 -31	0 -15	
Поля IP - пакета	Версия	IHL	Тип обслуживания	Длина дата граммы	Идентификация	Флаги	Смещение фрагмента	Время жизни (TTL)	Протокол	Контрольная сумма заголовка	IP-адрес отправителя	IP-адрес получателя	Параметры	Данные (до 65535 минус заголовок)

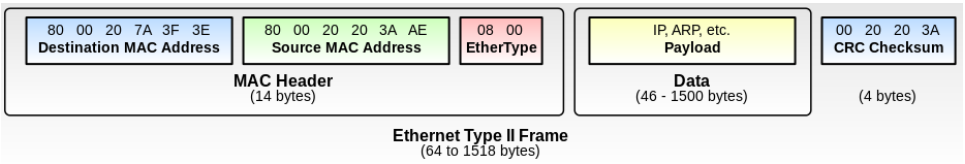
Рис. 3. Структура заголовка IP-пакета.

- Version (версия) – версия используемого протокола (IPv4 или IPv6).
- IHL (Internet Header Length) – длина заголовка IP-пакета в 32-битовых словах. Обычно заголовок имеет длину в 20 байт (пять 32-битовых слов).
- Type of Service (тип обслуживания) – уровень приоритета для данного IP-пакета.
- Datagram Length (длина дейтаграммы) – длина IP-пакета в целом. Длина раздела данных пакета вычисляется при помощи вычитания значения, хранящегося в поле IHL, из данного значения.
- Identification (идентификация) – Компьютер-отправитель помещает в это поле уникальный номер для каждого фрагмента сообщения. Для каждого пакета конкретного сообщении в данном 16-битовом поле будет содержаться одно и то же значение. Компьютер-получатель принимает все требуемые фрагменты, затем воссоздает исходное сообщение.
- Flags (флаги) – содержит различные флаговые биты. Нулевой бит

резервируется и всегда должен содержать нулевое значение. Первый бит определяет флаг DF (Don’t Fragment - не фрагментировать). Второй бит определяет флаг MF (More Fragments - дополнительные фрагменты).

- Fragment Offset (смещение фрагмента). – Если бит MF установлен равным единице, данное поле указывает позицию фрагмента в исходном сообщении, что обеспечивает успешную сборку пакета.
- TTL (Time To Live - время существования) – определяет время, в течение которого IP-пакет циркулирует в сети. При каждом прохождении пакета через маршрутизатор значение этого поля уменьшается на единицу и по достижении нуля пакет отбрасывается.
- Protocol (протокол) – содержит числовой код протокола, который присвоен ему организацией ICANN.
- Header Checksum (контрольная сумма) – содержит контрольную сумму, вычисляемую по всем полям заголовка IP-пакета. Это значение вычисляется повторно при каждом прохождении IP-пакета через маршрутизатор, поскольку при этом изменяется значение поля TTL.
- Source IP Address – содержит IP-адрес компьютера отправителя.
- Destination IP Address – содержит IP-адрес компьютера-получателя.
- Options (параметры) – дополнительные параметры, определяющие различные аспекты дальнейшей обработки пакета данных.
- Padding (дополнение) – дополняет заголовок, выравнивая его по 32-битной границе.

Максимальная длина поля данных IP-пакета ограничена разрядностью поля, определяющего эту величину, и составляет 65 535 байтов, однако при передаче по сетям различного типа длина пакета выбирается с учетом максимальной длины пакета протокола нижнего уровня, несущего IP-пакеты.



Если это кадры Ethernet, то выбираются пакеты с максимальной длиной в 1500 байтов, умещающиеся в поле данных кадра Ethernet.

3.2. Управление фрагментацией на уровне IP

В функции уровня IP входит разбиение слишком длинного для конкретного типа составляющей сети сообщения на более короткие пакеты. В большинстве типов локальных и глобальных сетей определяется такое понятие как максимальный размер поля данных кадра или пакета, в которые должен инкапсулировать свой пакет протокол IP. Эту величину обычно называют максимальной единицей транспортировки - Maximum Transfer Unit, MTU.

Ограничение на максимальный размер кадра накладывается по нескольким причинам:

- Для уменьшения времени на повторную передачу в случае потери или неисправимого искажения пакета. Вероятность потерь растёт с увеличением длины пакета.
- Чтобы при полудуплексном режиме работы хост не занимал долгое время канал.
- Чем больше отправляемый пакет, тем больше ожидание отправления других пакетов, особенно в последовательных интерфейсах. Поэтому маленький MTU был актуален во времена медленных коммутируемых соединений.

- Малый размер и быстродействие сетевых буферов входящих и исходящих пакетов. Однако слишком большие буферы тоже ухудшают производительность.

Сети Ethernet имеют значение MTU, равное 1500 байт, сети FDDI - 4096 байт, а сети X.25 чаще всего работают с MTU в 128 байт. Для иллюстрации работы протокола IP по фрагментации пакетов в хостах и маршрутизаторах рассмотрим рис. 4. На этом рисунке введем обозначения К1, Ф1 и К2, Ф2 для канального и физического уровней сети 1 и сети 2, соответственно.

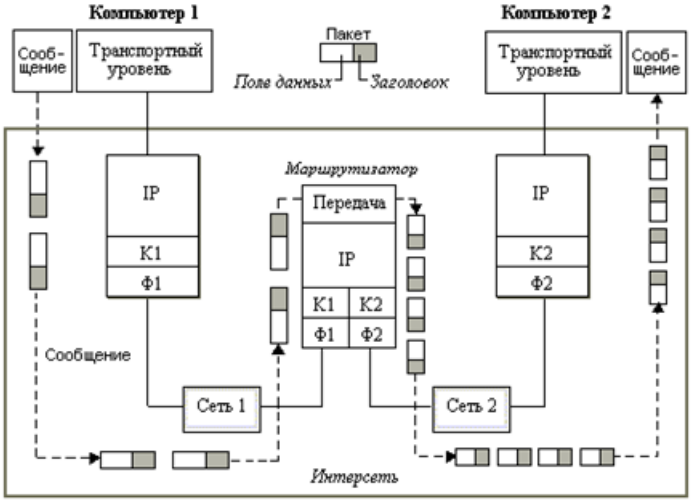


Рис. 4. Фрагментация IP-пакетов при передаче между сетями с разными MTU.

Пусть компьютер 1 подключен, например, к сети FDDI, имеющей значение MTU в 4096 байтов. При поступлении на его IP-уровень с транспортного уровня сообщения размером в 5600 байтов, протокол IP делит его на два IP-пакета, устанавливая в первом пакете признак фрагментации и присваивая пакету уникальный идентификатор, например, 486. В первом пакете величина поля смещения равна 0, а во втором - 2800. Признак фрагментации во втором пакете равен нулю, что показывает, что это последний фрагмент пакета. Общая величина IP-пакета составляет 2800+20 (размер заголовка IP), то есть 2820 байтов, что умещается в поле данных кадра FDDI.

Далее компьютер 1 передает эти пакеты на канальный (К1), а затем и на физический уровень (Ф1), который отправляет их маршрутизатору, связанному с данной сетью

Маршрутизатор по сетевому адресу определяет, что прибывшие два пакета надо передать в Ethernet сеть 2, которая имеет значение MTU, равное 1500. Маршрутизатор извлекает фрагмент транспортного сообщения из каждого пакета FDDI и делит его еще пополам, чтобы каждая часть уместилась в поле данных кадра Ethernet. Затем он формирует новые пакеты IP, каждый из которых имеет длину 1400 + 20 = 1420 байтов, что меньше 1500 байтов, поэтому они нормально помещаются в поле данных кадров Ethernet.

В результате в компьютер 2 по сети Ethernet приходит четыре IP-пакета с общим идентификатором 486, что позволяет протоколу IP, работающему в компьютере 2, правильно собрать исходное сообщение. Если пакеты пришли не в том порядке, в котором были посланы, то смещение укажет правильный порядок их объединения.

Отметим, что IP-маршрутизаторы не собирают фрагменты пакетов в более крупные пакеты, даже если на пути встречается сеть, допускающая такое укрупнение. Это связано с тем, что отдельные фрагменты сообщения могут перемещаться по интерсети по различным маршрутам, поэтому нет гарантии, что все фрагменты проходят через какой-либо промежуточный маршрутизатор на их пути.

Спецификация IP как не ограничивает дейтаграммы до маленького размера, так и не гарантирует, что большие дейтаграммы будут доставлены без фрагментации. Протокол IP устанавливает, что шлюзы должны принимать дейтаграммы с размерами, не более MTU сетей, к которым они присоединены и всегда должны всегда обрабатывать дейтаграммы до 576 октетов.

3.3. Основные особенности IP

- IP работает без создания логических соединений между хостами: он использует адреса, помещенные в заголовок IP пакета, для передачи их получателям. Выбор пути передачи называется маршрутизацией.
- IP использует поля в заголовке для фрагментации и восстановления дейтаграмм Internet, когда это необходимо для их передачи через сети с малым размером пакетов;
- Не требует подтверждения получения данных. Это означает, что отправитель и получатель не информируются о пропаже пакета или неправильной последовательности получения пакетов.

Если IP определит, что адрес получателя – локальный, то пакет отправляется непосредственно на хост назначения. В противном случае проверяется таблица маршрутизации на наличие маршрута к хосту назначения. Если маршрут найден, то пакет пересылается по найденному маршруту. В противном случае пакет пересылается на default gateway.

После получения маршрутизатором пакета, он передается на обработку IP, который выполняет следующие действия:

- Уменьшает значение TTL. Если TTL=0, то пакет уничтожается;
- Фрагментирует пакет на более мелкие, если пакет слишком большой для передачи по сети;
- Если пакет фрагментируется, то IP создает новые заголовки для каждого нового пакета, которые включают в себя:
  - Флаг означающий, что применялась фрагментация
  - Флаг показывающий, что это не последний фрагмент пакета.
  - Смещение фрагмента внутри пакета
- Пересчитывает контрольную сумму;
- Определяет адрес следующего маршрутизатора;
- Осуществляется пересылка пакета.

После прибытия пакета получателю IP собирает все фрагменты в единое целое.

(начало)

4. Транспортный уровень

Основными протоколами транспортного уровня являются протокол управления передачей TCP (Transmission Control Protocol) и протокол пользовательских дейтаграмм UDP (User Datagram Protocol). TCP обеспечивает надёжную доставку данных со сквозным обнаружением и устранением ошибок. UDP - это нетребовательная к ресурсам служба доставки дейтаграмм, работающая без образования логических соединений. Оба протокола выполняют доставку данных между прикладным и межсетевым уровнем. Разработчики программ сами выбирают, какая из этих служб точнее отвечает их задачам в каждом конкретном случае.

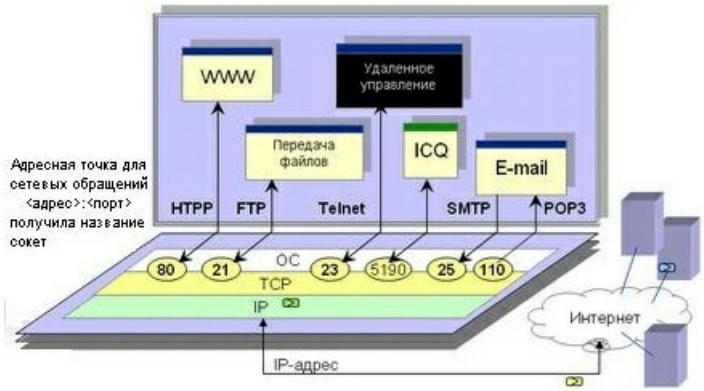


Рис. 5. Взаимодействие прикладных процессов в сети.

В то время как задачей сетевого уровня является передача данных между произвольными узлами сети, задача транспортного уровня заключается в передаче данных между любыми прикладными процессами, выполняющимися на любых узлах сети. Действительно, после того, как пакет средствами протокола IP доставлен компьютеру-получателю, данные необходимо направить конкретному процессу-получателю (рис. 5).

Каждый компьютер может выполнять несколько процессов, более того, прикладной процесс тоже может иметь несколько точек входа, выступающих в качестве адреса назначения для пакетов данных. Пакеты, поступающие на транспортный уровень, организуются операционной системой в виде множества очередей к точкам входа различных прикладных процессов.

В терминологии TCP/IP такие системные очереди называются портами. Таким образом, адресом назначения, который используется на транспортном уровне, является идентификатор (номер) порта прикладного сервиса. Номер порта, задаваемый транспортным уровнем, в совокупности с номером сети и номером сетевого интерфейса компьютера, однозначно определяют прикладной процесс в сети. Назначение номеров портов прикладным процессам осуществляется:

- либо централизовано, если эти процессы представляют собой популярные общедоступные сервисы, типа сервиса доступа к всемирной паутине WWW или сервиса удаленного управления Telnet,
- либо локально для тех сервисов, которые еще не стали столь распространенными, чтобы за ними закреплять стандартные (зарезервированные) номера.

Централизованное присвоение сервисам номеров портов выполняется организацией Internet Assigned Numbers Authority. Эти номера закрепляются и публикуются в стандартах Internet. Например, сервису WWW присвоен стандартный номер порта 80.

Локальное присвоение номера порта заключается в том, что разработчик некоторого приложения просто связывает с ним любой доступный, произвольно выбранный числовой идентификатор, обращая внимание на то, чтобы он не входил в число зарезервированных номеров портов. В дальнейшем все удаленные запросы к данному приложению от других приложений должны адресоваться с указанием назначенного ему номера порта.

4.1. Протокол доставки пользовательских дейтаграмм UDP

Это простой, компактный и очень быстродействующий протокол. Его IP-идентификатор равен 17 (0x11). Задачей протокола транспортного уровня UDP является передача данных между прикладными процессами без гарантии их доставки. Он не устанавливает логического соединения, не нумерует и не упорядочивает пакеты.

Протокол не гарантирует, что в процессе передачи его дейтаграммы не могут

быть потеряны, продублированы или прийти не в том порядке, в котором они были отправлены. Термин без гарантии доставки говорит о том, что средствами самого протокола невозможно убедиться, что данные корректно получены адресатом. В пределах одного компьютера UDP выполняет доставку данных безукоризненно.

С другой стороны, функциональная простота протокола UDP обуславливает простоту его алгоритма, компактность и высокое быстродействие. Поэтому те приложения, где реализован собственный, достаточно надежный, механизм обмена сообщениями, предпочитают для непосредственной передачи данных по сети использовать менее надежные, но более быстрые средства транспортировки.

Протокол UDP используют и тогда, когда хорошее качество связи обеспечивает достаточный уровень надежности и без применения дополнительных приемов типа установления логического соединения и квитирования передаваемых пакетов.

4.2. Формат сообщений UDP

Единица данных протокола UDP называется UDP-пакетом или пользовательской дейтаграммой. UDP-пакет состоит из заголовка и поля данных, в котором размещается пакет прикладного уровня (рис. 6).

Биты	0 - 15	16 - 31	32 -47	48 - 63	64 - …
Поля UDP - пакета	Порт отправителя (Source port)	Порт получателя (Destination port)	Длина дейтаграммы (Length)	Контрольная сумма (Checksum)	Данные (Data)

Рис. 6. Структура дейтаграммы UDP.

Первые 8 байт – UDP-заголовок, далее – данные сообщения. Значение поля Длина дейтаграммы указывает на длину всего UDP-сообщения, то есть, включая и UDP-заголовок. Измеряется в байтах.

Не все поля UDP-пакета обязательно должны быть заполнены. Если посылаемая дейтаграмма не предполагает ответа, то на месте адреса отправителя могут помещаться нули. Можно отказаться и от подсчета контрольной суммы. Однако следует учесть, что протокол IP подсчитывает контрольную сумму только для заголовка IP-пакета, игнорируя данные.

При вычислении максимальной длины данных в UDP-дейтаграмме необходимо принимать во внимание, что она в свою очередь является содержимым области данных IP-пакета, максимальная длина которого 65535 байт. Потому максимальная длина UDP-дейтаграммы за вычетом минимального IP-заголовка будет 65535 - 20 = 65515 байт (рис. 7).

Октеты	IP-сообщение	
65535	20	Минимальный IP-заголовок
	65515	Данные IP-сообщения:

UDP-сообщение	
8	UDP-заголовок
65507	Данные UDP-сообщения

Рис. 7. Включение UDP-дейтаграммы в структуру IP-пакета.

Если учесть, что длина заголовка 8 байт, то стало быть, максимальная длина данных в UDP-сообщении равна 65515 - 8 = 65507 байт. На практике невозможно использовать максимальную величину IP пакета, так как такой размер превышает максимального размера блока (в байтах), который может быть передан на канальном уровне.

UDP используется:

- если объём передаваемых данных столь невелик, что затраты на создание соединений и отслеживание надежности доставки будут больше, чем затраты на повторную передачу всех данных;
- для приложений, работающих по принципу запрос-ответ: ответ можно считать подтверждением приёма запроса,
- в приложениях, где есть собственные методы обеспечения надежной доставки данных и не требуется этого от транспортных протоколов..

4.3. Протокол надежной доставки TCP

Для обеспечения надёжной доставки данных на транспортном уровне используется протокол TCP – надёжный потоковый протокол, требующий создания логических соединений. Надёжность TCP обеспечивает механизм подтверждения приёма с повторной передачей (Positive Acknowledgment with Retransmission, PAR). Система с PAR повторяет отправку данных до тех пор, пока не получит от адресата подтверждение, что данные получены.

В протоколе TCP так же, как и в UDP, для связи с прикладными процессами используются порты, но если UDP действует по принципу «бросил письмо в почтовый ящик и забыл», то TCP требует «уведомления о получении заказного письма». Причем, достаточно одного уведомления о доставке целой группы IP-пакетов.

Данные представляются как поток байтов и могут передаваться в обоих направлениях. Единицей обмена данными для взаимодействующих модулей TCP является сегмент. Каждый сегмент имеет уникальный номер в последовательности (потоке) и содержит контрольную сумму, посредством которой получатель определяет целостность данных. Если сегмент данных получен в целости и сохранности, получатель отправляет источнику подтверждение (ACK). Повреждённые сегменты данных получателем игнорируются. По истечении установленного интервала ожидания (timeout) модуль-источник TCP повторно выполняет передачу всех сегментов, для которых не были получены подтверждения.

4.4. Конечные точки соединения и установление TCP-соединений

Работая на транспортном уровне, TCP дает возможность нескольким приложениям параллельно и независимо обмениваться данными с приложениями, запущенными на других машинах. Так же, как и UDP, протокол TCP демультиплексирует входной трафик между прикладными процессами и использует понятие порт, для идентификации конкретного процесса - получателя информации.

Для каждого соединения назначаются конечные точки соединения. Абстрактные объекты, представляющие конечные точки соединения, называют сокетами. Сокет - это программный интерфейс для обеспечения обмена данными между процессами. Различают клиентские и серверные сокеты. Клиентские сокеты можно сравнить с оконечными аппаратами телефонной сети, а серверные - с телефонными коммутаторами. Под конечными точками соединения понимается пара целых чисел вида:

```
< узел > : < порт > ,
```

где <узел> - определяет IP-адрес узла сети, а <порт> - номер порта TCP данного узла. Так, запись 95.140.148.234:80 определяет конечную точку, которая характеризуется TCP-портом с номером 80, относящимся к компьютеру с IP-адресом 95.140.148.234.

Любое соединение идентифицируется парой его конечных точек. При этом одна и та же конечная точка может использоваться несколькими соединениями. Неоднозначности не возникает, так как в протоколе TCP все соединения связаны с открытым соединением, которое определено парой конечных точек, а не номерами портов. Большинство ОС поддерживают возможность одновременного доступа нескольких клиентов к службе электронной почты. Благодаря этому несколько клиентов могут отправлять почтовые сообщения одновременно. Так как программа, принимающая входящие сообщения, использует для связи протокол TCP, то для ее работы достаточно назначить лишь один локальный порт, несмотря на то, что могут одновременно обрабатываться несколько соединений.

Протокол TCP требует установку соединения, то есть предварительную "договоренность" между конечными точками двух компьютеров. Эти "переговоры" ведут приложения, запущенные на компьютерах между которыми устанавливается соединение:

- Одно из приложений реализует функцию пассивного открытия соединения (Listen), сообщая, что оно готово к приему входящих соединений. После этого операционная система назначает номер TCP-порта для собственной конечной точки текущего соединения.
- Приложение, которое выполняется на другом конце виртуального канала, обращается к операционной системе с запросом на активное открытие соединения (Connect). Два экземпляра протокола TCP взаимодействуют друг с другом.
- В ходе этого взаимодействия проверяется возможность установки соединения. После установки соединение, приложения начинают обмен данными. При этом экземпляры TCP в фоновом режиме обмениваются сообщениями, гарантирующими надежность доставки.

Протокол TCP разбивает поток данных на сегменты, каждый из которых, как правило, передается в сети в виде единственной IP-дейтаграммы.

4.5. Структура TCP-сегментов

В протоколе TCP для установки соединения, передачи данных, отправки сигналов подтверждения приема, объявления размера окон и закрытия соединения используются сегменты, формат которых приведен на рис. 8

Рис. 8. Структура сегмента TCP.

Каждый сегмент состоит из заголовка и раздела данных. В заголовке идентификационные данные и управляющая информация выделены в ряд полей, а именно:

- Source port – порт компьютера-отправителя.
- Destination port – порт компьютера-получателя.

- Sequence number (Порядковый номер) – используют для сборки фрагментов в одно большое сообщение после получения IP-пакета.
- Acknowledgment number (Номер подтверждения) – хранит сведения о номере подтверждения приема IP-пакета.
- Data offset (Смещение данных) – определяет место раздела данных в IP-пакете.
- Reserved – зарезервировано для использования в будущем.
- Code bits (Кодовые биты) – информация о типе данного сегмента, задаваемая установкой в единицу соответствующих бит этого поля:
  - URG - срочное сообщение;
  - ACK - квитанция на принятый сегмент;
  - PSH - запрос на отправку без ожидания заполнения буфера;
  - RST - запрос на восстановление соединения;
  - SYN – сообщение, используемое для синхронизации счетчиков переданных данных при установлении соединения;
  - FIN - признак достижения передающей стороной последнего байта в потоке передаваемых данных.
- Window (Окно) – количество блоков, принимаемых получателем.
- Checksum (Контрольная сумма) – для контроля целостности разделов, содержащих заголовок и данные пакета.
- Urgent pointer (Указатель срочности) – с битом URG указывает на конец данных, которые необходимо срочно принять, несмотря на переполнение буфера
- Options (Параметры) – имеет переменную длину и используется для вспомогательных задач. Например, при выборе максимального размера сегмента.
- Padding – фиктивное поле, используемое для доведения размера заголовка до целого числа 32-битовых слов

4.6. Порядок установки TCP-соединений

В TCP данные считаются непрерывным потоком байтов, а не набором независимых пакетов. Следовательно, TCP предпринимает меры для сохранения последовательности отправки и получения байтов. Этой цели служат поля заголовка TCP – Порядковый номер и Номер подтверждения.

Чтобы корректно отслеживать порядок в потоке данных, каждая из взаимодействующих сторон должна знать исходный номер второй стороны. Две стороны соединения синхронизируют системы нумерации байтов, обмениваясь SYN-сегментами в ходе рукопожатия. Соединение инициализируется путем трех обменов данными (three-way handshake – тройное рукопожатие):

Рис. 9. Метод установки сеанса связи TCP.

1. Инициализация запроса на соединения путем посылки сегмента с

установленным флагом синхронизации (SYN);

2. Подтверждение установки соединения хостом получателем путем посылки сегмента, содержащего:
   - Включенный флаг синхронизации (SYN);
   - Число, определяющее стартовый номер последовательности сегментов, которую хост будет отправлять;
   - Подтверждение, содержащее номер стартового сегмента получаемых данных;
3. Хост, инициирующий соединения, посылает подтверждение приема данных.

Аналогичная процедура происходит при закрытии соединения: тройное рукопожатие с байтом FIN. Именно сквозной обмен данными становится логическим соединением между двумя хостами.

(начало)

5. Приложение. Протокол надежной доставки сообщений TCP

В стеке протоколов TCP/IP протокол *TCP (Transmission Control Protocol)* работает так же, как и протокол UDP, на транспортном уровне. Он обеспечивает надежную транспортировку данных между прикладными процессами путем установления логического соединения.

5.1. Понятия сегмента протокола TCP

Единицей данных протокола TCP является сегмент. Информация, поступающая к протоколу TCP в рамках логического соединения от протоколов более высокого уровня, рассматривается протоколом TCP как неструктурированный поток байт. Поступающие данные буферизуются средствами TCP. Для передачи на сетевой уровень из буфера "вырезается" некоторая непрерывная часть данных, называемая сегментом.

В протоколе TCP предусмотрен случай, когда приложение обращается с запросом о срочной передаче данных (бит PSH в запросе установлен в 1). В этом случае протокол TCP, не ожидая заполнения буфера до уровня размера сегмента, немедленно передает указанные данные в сеть. О таких данных говорят, что они передаются вне потока - *out of band.*

Не все сегменты, посланные через соединение, будут одного и того же размера, однако оба участника соединения должны договориться о максимальном размере сегмента, который они будут использовать. Этот размер выбирается таким образом, чтобы при упаковке сегмента в IP-пакет он помещался туда целиком, то есть максимальный размер сегмента не должен превосходить максимального размера поля данных IP-пакета. В противном случае пришлось бы выполнять фрагментацию, то есть делить сегмент на несколько частей, для того, чтобы он вместился в IP-пакет.

Аналогичные проблемы решаются и на сетевом уровне. Для того, чтобы избежать фрагментации, должен быть выбран соответствующий максимальный размер IP-пакета. Однако при этом должны быть приняты во внимание максимальные размеры поля данных кадров (MTU) всех протоколов канального уровня, используемых в сети. Максимальный размер сегмента не должен превышать минимальное значение на множестве всех MTU составной сети.

5.2. Порты и установление TCP-соединений

В протоколе TCP также, как и в UDP, для связи с прикладными процессами используются порты. Номера портам присваиваются аналогичным образом:

имеются стандартные, зарезервированные номера (например, номер 21 закреплен за сервисом FTP, 23 - за telnet), а менее известные приложения пользуются произвольно выбранными локальными номерами.

Однако в протоколе TCP порты используются несколько иным способом. Для организации надежной передачи данных предусматривается установление *логического соединения* между двумя прикладными процессами. В рамках соединения осуществляется обязательное подтверждение правильности приема для всех переданных сообщений, и при необходимости выполняется повторная передача. Соединение в TCP позволяет вести передачу данных одновременно в обе стороны, то есть полнодуплексную передачу.

Соединение в протоколе TCP идентифицируется парой полных адресов обоих взаимодействующих процессов (оконечных точек). Адрес каждой из оконечных точек включает IP-адрес (номер сети и номер компьютера) и номер порта. Одна оконечная точка может участвовать в нескольких соединениях.

Установление соединения выполняется в следующей последовательности:

- При установлении соединения одна из сторон является инициатором. Она посылает запрос к протоколу TCP на открытие порта для передачи (active open).
- 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 (passive open) и возвращает квитанцию, подтверждающую прием запроса.
- Для того чтобы передача могла вестись в обе стороны, протокол на приемной стороне также открывает порт для передачи (active port) и также передает запрос к противоположной стороне.
- Сторона-инициатор открывает порт для приема и возвращает квитанцию. Соединение считается установленным. Далее происходит обмен данными в рамках данного соединения.

5.3. Концепция квитирования

В рамках соединения правильность передачи каждого сегмента должна подтверждаться квитанцией получателя. *Квитирование* - это один из традиционных методов обеспечения надежной связи. Идея квитирования состоит в следующем.

Для того, чтобы можно было организовать повторную передачу искаженных данных отправитель нумерует отправляемые единицы передаваемых данных (далее для простоты называемые кадрами). Для каждого кадра отправитель ожидает от приемника так называемую положительную квитанцию - служебное сообщение, извещающее о том, что исходный кадр был получен и данные в нем оказались корректными. Время этого ожидания ограничено - при отправке каждого кадра передатчик запускает таймер, и если по его истечению положительная квитанция на получена, то кадр считается утерянным. В некоторых протоколах приемник, в случае получения кадра с искаженными данными должен отправить отрицательную квитанцию - явное указание того, что данный кадр нужно передать повторно.

Существуют два подхода к организации процесса обмена положительными и отрицательными квитанциями: с простоями и с организацией "окна".

Метод с простоями требует, чтобы источник, пославший кадр, ожидал получения квитанции (положительной или отрицательной) от приемника и только после этого посылал следующий кадр (или повторял искаженный). Из рис. 10 видно, что в этом случае производительность обмена данными существенно снижается - хотя передатчик и мог бы послать следующий кадр

сразу же после отправки предыдущего, он обязан ждать прихода квитанции. Снижение производительности для этого метода коррекции особенно заметно на низкоскоростных каналах связи, то есть в территориальных сетях.

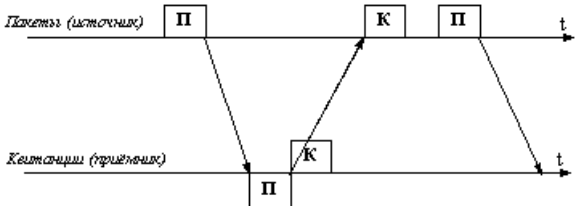


Рис. 10. Метод подтверждения корректности передачи кадров с простоем источника.

Во втором методе для повышения коэффициента использования линии источнику разрешается передать некоторое количество кадров в непрерывном режиме, то есть в максимально возможном для источника темпе, без получения на эти кадры ответных квитанций. Количество кадров, которые разрешается передавать таким образом, называется размером окна. Рисунок 11 иллюстрирует данный метод для размера окна в W кадров. Обычно кадры при обмене нумеруются циклически, от 1 до W. При отправке кадра с номером 1 источнику разрешается передать еще W-1 кадров до получения квитанции на кадр 1. Если же за это время квитанция на кадр 1 так и не пришла, то процесс передачи приостанавливается, и по истечению некоторого тайм-аута кадр 1 считается утерянным (или квитанция на него утеряна) и он передается снова.

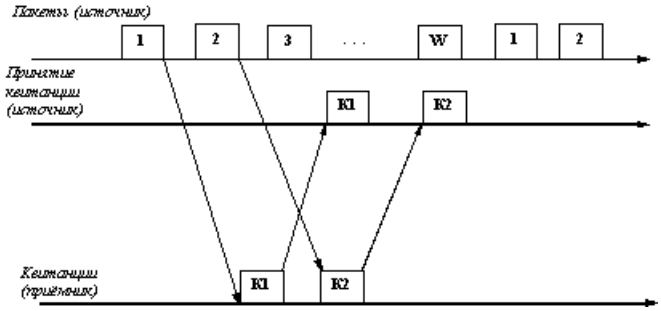


Рис. 11. Метод "окна" - непрерывная отправка пакетов.

Если же поток квитанций поступает более-менее регулярно, в пределах допуска в W кадров, то скорость обмена достигает максимально возможной величины для данного канала и принятого протокола.

Этот алгоритм называют алгоритмом скользящего окна. Действительно, при каждом получении квитанции окно перемещается (скользит), захватывая новые данные, которые разрешается передавать без подтверждения.

5.4. Реализация скользящего окна в протоколе TCP

В протоколе TCP реализована разновидность алгоритма квитирования с использованием окна. Особенность этого алгоритма состоит в том, что, хотя единицей передаваемых данных является сегмент, окно определено на множестве нумерованных байт неструктурированного потока данных, поступающих с верхнего уровня и буферизуемых протоколом TCP.

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.

В качестве квитанции получатель сегмента отсылает ответное сообщение (сегмент), в которое помещает число, на единицу превышающее максимальный номер байта в полученном сегменте. Если размер окна равен W,

а последняя квитанция содержала значение N, то отправитель может посылать новые сегменты до тех пор, пока в очередной сегмент не попадет байт с номером N+W. Этот сегмент выходит за рамки окна, и передачу в таком случае необходимо приостановить до прихода следующей квитанции.

5.5. Выбор тайм-аута

Выбор времени ожидания (тайм-аута) очередной квитанции является важной задачей, результат решения которой влияет на производительность протокола TCP.

Тайм-аут не должен быть слишком коротким, чтобы по возможности 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 несуществующей или "заблудившейся" квитанции.

При выборе величины тайм-аута должны учитываться скорость и надежность физических линий связи, их протяженность и многие другие подобные факторы. В протоколе TCP тайм-аут определяется с помощью достаточно сложного адаптивного алгоритма, идея которого состоит в следующем. При каждой передаче засекается время от момента отправки сегмента до прихода квитанции о его приеме (время оборота). Получаемые значения времен оборота усредняются с весовыми коэффициентами, возрастающими от предыдущего замера к последующему. Это делается с тем, чтобы усилить влияние последних замеров. В качестве тайм-аута выбирается среднее время оборота, умноженное на некоторый коэффициент. Практика показывает, что значение этого коэффициента должно превышать 2. В сетях с большим разбросом времени оборота при выборе тайм-аута учитывается и дисперсия этой величины.

5.6. Реакция на перегрузку сети

Варьируя величину окна, можно повлиять 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 узлах-маршрутизаторах и в конечных узлах-компьютерах.

При переполнении приемного буфера конечного узла "перегруженный" протокол TCP, отправляя квитанцию, помещает в нее новый, уменьшенный размер окна. Если он совсем отказывается от приема, то в квитанции указывается окно нулевого размера. Однако даже после этого приложение может послать сообщение на отказавшийся от приема порт. Для этого, сообщение должно сопровождаться пометкой "срочно" (бит URG в запросе установлен в 1). В такой ситуации порт обязан принять сегмент, даже если для этого придется вытеснить из буфера уже находящиеся там данные.

После приема квитанции с нулевым значением окна протокол-отправитель время от времени делает контрольные попытки продолжить обмен данными. Если протокол-приемник уже готов принимать информацию, то в ответ на контрольный запрос он посылает квитанцию с указанием ненулевого размера окна.

Другим проявлением перегрузки сети является переполнение буферов в маршрутизаторах. В таких случаях они могут централизовано изменить размер окна, посылая управляющие сообщения некоторым конечным узлам, что позволяет им дифференцировано управлять интенсивностью потока данных в разных частях сети.