Тема8 Транспортний рівень

Послуги, які надаються верхнім рівням.

Функции транспортного протокола двояка.

С одной стороны, транспортный протокол определяет средства, необходимые для взаимодействия процессов, - транспортный интерфейс.

2. С другой стороны, этот протокол организует сопряжение процессов и, - ЭВМ с сетью, построенной по правилам, определяемым сетевым протоколом X25.

На транспортный протокол возлагают функции:

- обеспечивают надежную связь для передачи данных между взаимодействующими процессами;
- создания соединения между портами процессов (порты создаются на сеансовом уровне;
- передача сообщений через логические соединения;
- обнаружение сбоев и восстановление информации;
- обнаружение дубликатов пакетов и упорядочение пакетов;
- задача фрагментации сообщений, т.е. разбивка сообщений на пакеты оптимальной длины;
- управление протоколами и буферизация данных;
- организация приоритетных передач пакетов, что связано с необходимостью передачи управляющих пакетов для быстрого реагирования на изменения в сети;
- защита передачи данных;
- инициализация и восстановление состояния отказа, что связано с управлением работой сети для восстановления работоспособности сети после сбоев.

На транспортном уровне данные передаются в виде сообщений мах длины 8Кбайт. Сообщения разбиваются на пакеты (Заголовок + Блок данных пользователя).

Заголовок имеет переменную длину, - содержит:

Указатель длины заголовка,

Адрес порта местного процесса, передающего пакет,

Адрес порта удаленного процесса, принимающего пакет,

Порядковый N пакета в сеансе связи.

Транспортный уровень - предоставляет услуги уровню сеансов. Сети связи, используемые для передачи данных, могут быть разных типов и форм (с коммутацией пакетов, каналов, локальной, региональной или объединением сетей всех этих типов; в них могут входить относительно медленные спутниковые каналы или наземные каналы с помехами).

Задачей транспортного уровня является ограждение сеансового уровня от капризов сетевых механизмов, лежащих в его основе. Услуга, которую он предоставляет

вышестоящему сетевому уровню, это надежный и прозрачный механизм передачи данных. Этот механизм описывается на языке требований к качеству обслуживания.

Или

Единственной целью транспортного уровня - обеспечить эффективный, надежный и дешевый сервис для пользователей на прикладном уровне. Для этого он использует сервис, предоставляемый сетевым уровнем. То что выполняет работу транспортного уровня называется транспортным агентом. Транспортный агент может располагаться в ядре операционной системы, в отдельном процессе пользователя, в библиотеке сетевого приложения, на карте сетевого интерфейса. В некоторых случаях оператор сети может предоставлять надежный транспортный сервис, при котором транспортный агент располагается на специальной интерфейсной машине на границе подсети, к которой подключены хосты. На рис. 6-1 показаны взаимно расположения сетевого, транспортного и прикладного уровня.

Подобно тому, как сетевой уровень имеет два вида сервиса: ориентированный на соединения и нет, транспортный так же имеет сервис, ориентированный на соединения и без соединений. Адресация и управление потоком схожи на обоих уровнях.

Тогда возникает вопрос: Если сервис сетевого уровня столь схож с сервисом транспортного, то зачем два разных уровня? Причина этого состоит в том, что сетевой уровень - это часть подсети передачи данных, которой управляет оператор подсети. Что будет если сетевой уровень предоставляет ненадежный сервис, ориентированный на соединения? Предположим, что он часто теряет пакеты? Что делать если маршрутизатор время от времени отказывает?

У пользователя подсетью нет средств для решения проблемы если она возникла. Для того, чтобы ему дать такую возможность надо поверх сетевого пустить еще один уровень, который позволит исправлять качество сетевого уровня. Если транспортному уровню придет сообщение, что соединение на сетевом уровне неожиданно было разорвано, то он может установить новое сетевое соединение с помощью которого выяснить что произошло, какие данные были переданы, а какие нет и т.п.

Существо транспортного уровня в том, чтобы сделать сервис транспортного уровня более надежным, чем сетевого. Другое важное соображение в том, что прикладная программа, опираясь на транспортный сервис, становится независимой от сети и может работать в сети с любым сервисом.

В силу приведенных доводов первые четыре уровня называют поставщиками транспортного сервиса, а все что выше четвертого - пользователями транспортного сервиса.

Примітиви транспортної служби

Примитивы являются абстрактными представлениями взаимодействий через точки доступа услуг, которые указывают, что между пользователем и поставщиком услуги передается информация. Для соединения 4 типа примитивов:

- запрос
- признак
- ответ

- подтверждение

Примитивы транспортного уровня :1) запрос и ответ выбирается пользователем услуги; 2) признак и подтверждение выбирается поставщиком услуг.

Функціонування транспортної служби

Функционирование транспортной службы

1. Началом работы **транспортной службы** служит передача активизируемым процессов в местную транспортную службу примитива "соединения" . В нем указывается:

- адрес удаленного процесса
- определяется вид требуемой **транспортной** услуги

Этот примитив поступает от пользователя транспортного соединения, которые, как правило, является некоторым объектом представительского уровня.

2. В ответ на это транспортный элемент местной **транспортной службы** формирует Блок Данных Транспортного Протокола и пересылает его транспортному объекту получателя. Получатель после поступления этого блока оповещает о запросе на соединения указ. получателя.

3. **Если получатель готов принять вызов , то он выдает примитив "согласия". Затем оба транспортных объекта обмениваются сообщениями для построения блоков связи, которые определяют устанавливаемое соединение**. Блоки связи формируются в специальные области памяти, доступ к которым получает транспортная служба и процесс.

После этого логическое соединение считается установленным и осуществляется передача данных в соответствии с заданным режимом обмена. По окончанию сеанса обмена реализуется процедура разъединения логического соединения.

Класи транспортних протоколів

1. Классы транспортных протоколов:

0 - примитивный класс (очень простое транспортное соединение)

1 - базовый класс. Восстановление от ошибок (сети X.25.). X.25 не могут справится со всеми проблемами, транспортный уровень выполняет важную задачу обеспечиваемой сквозной целостности сети.

2 - класс мультипликации называется мультипликацией нескольких транспортных соединений в одном сеансе связи сети X.25, используется в сетях класс A, управление

протоколом осуществляется в соответствии с принципом "окна".

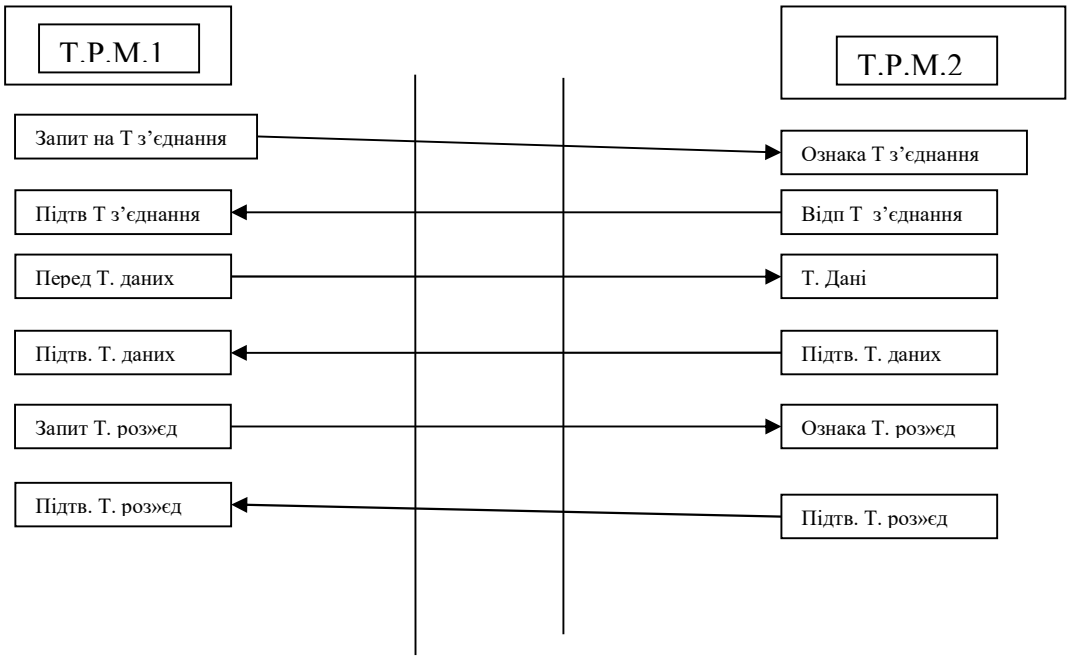
3 - то же что и класс 2.

восстановление от ошибок и мультипликация, обслуживание сетей типа B, обеспечение восстановление от сбоев без требования уведомления пользователя.

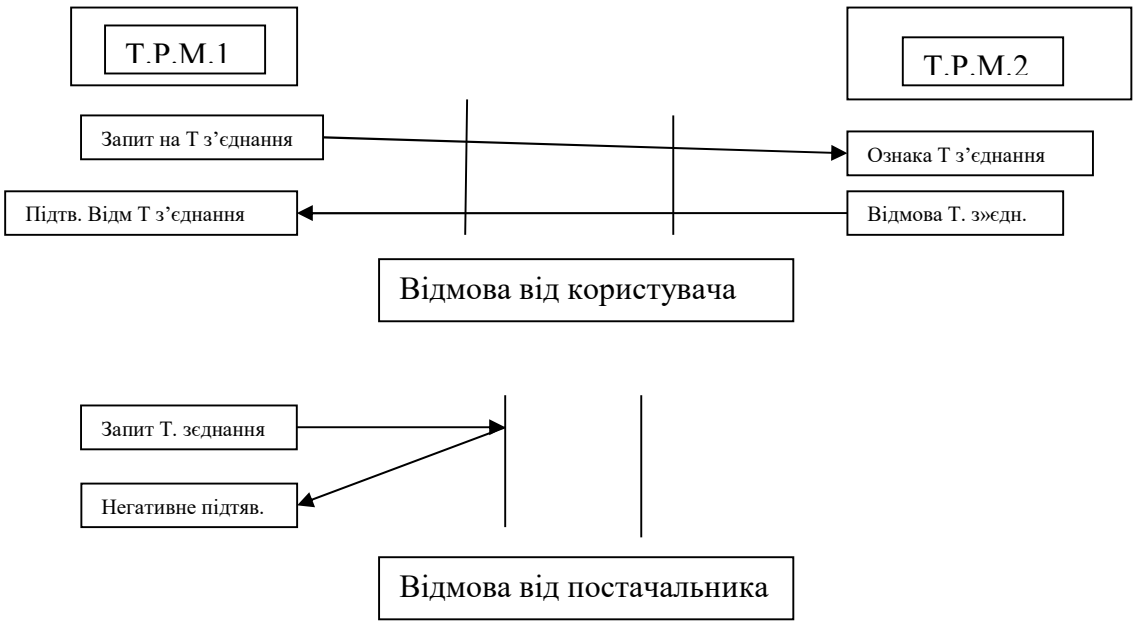
4 - включает функции управления протоколом классов 2 и 3, класс обнаружения ошибок и восстановление от ошибок. Используется ряд сложных механизмов для проверки ошибок, решение проблем последних номеров и потерянных пакетов.

Клас транспортного протоколу	Тип мережі
TP0	A
TP1	B
TP2	A
TP3	B
TP4	C

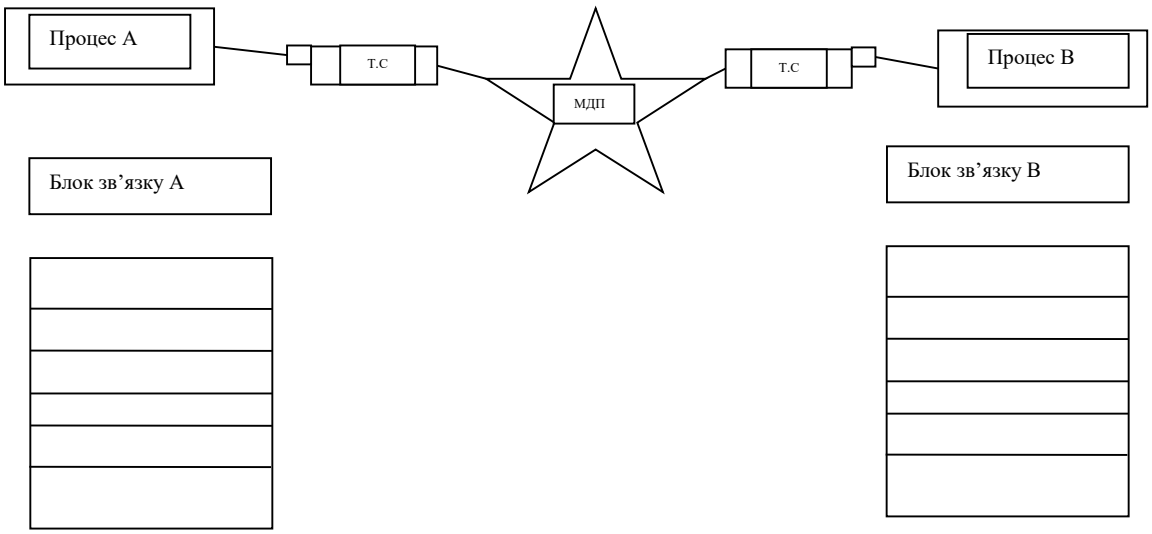
Транспортне з'єднання



Існує можливість нереалізації даного транспортного з’єднання, якщо параметри передачі чи якість обслуговування, сформовані користувачем ініціаторм з’єднання є надмірними і не можуть бути реалізовані в даному т. з»єданні.



Блоки зв’язку



Блоки связи формируются в специальные области памяти, доступ к которым получает транспортная служба и процесс.

Блоки связи должен создать параметры соединения:

- адрес порта местной **транспортной службы** (местного процесса) . C и D
- адрес местного процесса логическое устройства ввода/вывода. A и B
- адрес удаленного процесса
- адрес порта удаленной **транспортной службы**
- информация о соединении связи (установлена, разъединена, разорвана)
- параметры услуг связи для определения класса транспортного протокола - с их помощью реализуются поток сообщений
- размер передающего буфера
- размер приемного буфера
- номер следующего передаваемого сообщения N(S)
- номер следующего принимаемого сообщения N(R)
- max номер передаваемого сообщения M(S)
- max номер принимаемого сообщения M(R)

Тема9 Мережа Інтернет

Організація інформаційного обміну в мережі інтернет

IAB - Internet Architecture (Activities) Board

Включає 2 підкомітети:

- Дослідницький IRTF – Internet Research Task Force
- Законодачий IETF – Internet Engineering Task Force. Аналізує інформацію, надану IRTF і стандартизує, регламентує та публікує відкриті стандарти та рекомендації, які повинні використовуватися та застосовуватись користувачами Інтернет. Розробляє і публікує документи RFC (Request for comments). Всі питання адресації, розвитку мережі в цілому так і ф-ціонування відповідних протоклів.

NIC – Network information Center. Займається тільки орг. Питаннями, відповідає за розповсюдження тех.. документації . Проводить роботу з реєстрації та підключення нових користувачів, а також відповідає за розподіл адресного середовища і реєстрацію доменів.

Тема10 Структура та сервіси мережі інтернет

Структура та сервіси мережі інтернет

Структура Internet

Пять лет назад ответ был прост: Internet – это все сети, которые, взаимодействуя с помощью протокола IP, образуют «бесшовную» сеть для своих коллективных пользователей. Сюда относятся различные федеральные сети, совокупность региональных сетей, университетские сети и некоторые зарубежные сети.
В последнее время появилась заинтересованность в подсоединении к Internet сетей, которые не используют протокол IP. Для того чтобы предоставлять клиентам этих сетей услуги Internet, были разработаны методы подключения этих «чужих» сетей (например, BITNET, DECnets и др.) к Internet. Сначала эти подключения, названные шлюзами, предназначались просто для пересылки электронной почты между двумя сетями, но некоторые из них выросли до возможности обеспечения и других услуг на межсетевой основе. Являются ли они частью Internet? И да и нет – всё зависит от того, хотят ли они того сами.

Фактически Internet состоит из множества локальных и глобальных сетей, принадлежащих различным компаниям и предприятиям, связанных между собой различными линиями связи. Internet можно представить себе в виде мозаики сложенной из небольших сетей разной величины, которые активно взаимодействуют одна с другой, пересылая файлы, сообщения.

Типы сервисов Интернет
- Электронная почта
- Сетевые новости Usenet
- Списки рассылки
- FTP - передача файлов
- Система поиска файлов Archie
- Гипертекстовая система Gopher
- Система гипермедиа WWW
- Гипертекстовая система Hyper-G
- Поисковая система WAIS
- Сервисы IRC, MUD, MOO
- Инфраструктурные сервисы
- Современные разработки, использующие Интернет как среду передачи информации

Шлюзові протоколи

Шлюзовые протоколы EGP (Exterior Gateway Protocol), GGP (Gateway-to-gateway protocol), IGP (Interior Gateway Protocol) (RIP, HELLO) отвечают за передачу информации о маршрутизации данных и состоянии сети, а также обрабатывают данные для взаимодействия с локальными сетями;

Стек протоколів TCP/IP

- Відкриті стандарти, які розробляються незалежно від апаратного та програмного забезпечення
- Незалежність від фіз.. середовища передачі
- Унікальна система адресації
- Стандартизація протоколів вищіх рівнів для найбільш поширених сервісів користувачів.

  Первая проблема была связана с развитием программного обеспечения, способного объединить несколько сетей с разными оперативными системами. Вторая проблема заключалась в создании такого программного обеспечения, чтобы «сеть из сетей» могла продолжать функционирование даже в случае потери нескольких компьютеров. Решение этих двух проблем требовало огромного объема работы и талантливых специалистов, что, в конечном результате, привело к созданию программы TCP/IP.

  Среди преимуществ программы TCP/IP - ее крайне децентрализованная система. Ни правительство, ни корпорационные монополии не имеют контроль над ее работой. Соединение с Интернетом также не требует официального разрешения. Наоборот, как дикая земляника, ветви Интернета разрастаются горизонтально, демократично, в то время как новые региональные компьютерные службы Интернета (Internet Service Providers - ISPs) во всем мире покупают мощные серверы, устанавливают программу TCP/IP, подсоединяются к другому TCP/IP компьютеру сети и предоставляют доступ к Интернету отдельным лицам и местным организациям. Интернет развивается так быстро, что его рост измеряется

в процентах в месяц.Поначалу сеть помогала лишь ученым пользоваться информацией, находящейся в компьютерах коллег в других центрах. Тогда еще никому не приходило в голову, каких масштабов достигнет Интернет. Однако профессор не считает, что он вместе с коллегами породил монстра.

Порівняння стеку протоколів

Тема 11 Структура ІП-адреси та ІП-адресація

IP-адресація

IP-адреса – логічна адреса чи адреса п.з. станції

IPv4: ключ(1-5біт)-адр мережі(NET ID) – адр хост-вузла(HOST ID) (Всього 32 біти)

Клас мережі	Структура IP-адреси	Ключ	Dec значення 1-го байту	Кількість мереж	Кількість вузлів	Примітка
A	Net.Node.Node.Node	0	1~126	126 (2^7-2)	2^{24}-2	50%
B	Net.Net.Node.Node	10	128~191	2^{14}-2	2^{16}-2	25%
C	Net.Net.Net.Node	110	192~223	2^{21}-2	254	12.5%
D	----------	1110	224~239	----------	----------	Broadcast
E	----------	11110	240~247	----------	----------	Reserved

Виділення під мереж

Виділення під мережі – це процедура представлення 1 великої мережі як сукупності під мереж меншого розміру. При цьому кожна мережа повинна мати в IP – адресі свій ідентифікатор.

NET ID	SUBNET ID	Host ID

Ip-адреса

Розмір SubNetID залежить від кількості потрібних мереж.

Щоб визначити в яких розрядах IP адреси представлено ID мережі, під мережі, а в яких – хосту, вводять маску.

Маска – це 32 бітне слово, яке складається з послідовності «1» за якою іде послідовність «0». «1» в масці вказують на ті розряди IP-адреси, в яких знаходяться адреси мережі та під мережі. «0» - показують в яких розрядах IP-адреси занесено адресу хост-вузла.

Маски за замовчуванням:

A – 255.0.0.0

B – 255.255.0.0

C – 255.255.255.0

Поняття розширеного мережного префіксу

NetID	SubNet ID	Host ID
1111…..11	111111……..111	0000……..000
Префікс мережі		
Розширений префікс		

Тема12 Структура та функції основних протоколів

IPv4

Протокол IP (RFC 791) используется для негарантированной доставки данных (разделяемых на так называемые пакеты) от одного узла сети к другому. Это означает, что на уровне этого протокола (третий уровень сетевой модели OSI) не даётся гарантий надёжной доставки пакета до адресата. В частности, пакеты могут прийти не в том порядке, в котором были отправлены, продублироваться (когда приходят две копии одного пакета; в реальности это бывает крайне редко), оказаться повреждёнными (обычно повреждённые пакеты уничтожаются) или не прибыть вовсе. Гарантии безошибочной доставки пакетов дают протоколы более высокого (транспортного) уровня сетевой модели OSI — например, TCP — которые IP используют в качестве транспорта.

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; название «октет» идёт с тех времён, когда байты на разных компьютерах содержали разное число битов). При этом компьютеры в подсетях объединяются общими начальными битами адреса. Количество этих бит, общее для данной подсети, называется маской подсети (ранее использовалось деление пространства адресов по классам — A, B, C; класс сети определялся диапазоном значений старшего октета и определял число адресуемых узлов в данной сети, сейчас используется бесклассовая адресация).

В протоколе четвертой версии (IPv4)

0								1								2								3							
0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7	0	1	2	3	4	5	6	7
Версия				**IHL**				**Приорітет**			**D**	**T**	**R**	**C**	**X**	**Длина пакета**																
Идентификатор																**X**	**DF**		**MF**	**Смещение фрагмента**										
Число переходов (TTL)								**Протокол**								**Контрольная сумма заголовка**															
IP-адрес отправителя (32 бита)																															
IP-адрес получателя (32 бита)																															
Параметры (до 320 бит)																**Данные (до 65535 байт минус заголовок)**														

- Версия — для IPv4 значение поля должно быть равно 4.
- IHL — длина заголовка IP-пакета в 32-битных словах (dword). Именно это поле указывает на начало блока данных в пакете. Минимальное корректное значение для этого поля равно 5.
- Приоритет – показывает как будет обработана данная дейтаграмма
  - 0 – обычный
  - 1 – приоритетный
  - 2 - Неотложный
  - 3 – Срочный
  - 4 – экстренный
  - 5-9 – передача служебных сообщений
- D – передача с минимальными задержками
- T – передача с максимальной пропускной способностью
- R – передача с максимальной надежностью
- C – передача с минимальной ценой
- D T R C : 1 1 1 1 - передача с максимальной безопасностью
- Идентификатор — значение, назначаемое отправителем пакета и предназначенное для определения корректной последовательности фрагментов при сборке датаграммы. Для фрагментированного пакета все фрагменты имеют одинаковый идентификатор.
- 3 бита флагов.
  - Первый бит должен быть всегда равен нулю,
  - второй бит DF (don’t fragment) определяет возможность(0) или запрет (1) фрагментации пакета
  - третий бит MF (more fragments) показывает, не является ли этот пакет последним в цепочке пакетов(0).
- Смещение фрагмента — значение, определяющее позицию фрагмента в потоке данных. (64 битные слова)

- TTL (Time To Live) – показывает, сколько времени может передаваться в сети эта дейтаграмма. На каждом шагу от значения, установленного пользоваталем-отправителем отнимается 1. Как только поле=0, дейтаграммам уничтожается.
- Протокол — идентификатор интернет-протокола следующего уровня
- Контрольная сумма – определяет корректность передачи дейтаграммы
- Параметры – дополнительные опции передачи (временные х-ки, секретные ключи и т.д)

IPv6

IPv6 (англ. *Internet Protocol version 6*) — это новая версия протокола IP, призванная решить проблемы, с которыми столкнулась предыдущая версия (IPv4) при её использовании в Интернете, за счёт использования длины адреса 128 бит вместо 32. В настоящее время протокол IPv6 уже используется в нескольких сотнях сетей по всему миру (более 1600 сетей на март 2009), но пока ещё не получил столь широкого распространения в Интернете, как IPv4. Протокол был разработан IETF.

При этом к проблемам масштабируемости протокола IPv4 следует отнести следующие:

- недостаточность объёма 32-битного адресного пространства;
- сложность агрегирования маршрутов, разрастание таблиц маршрутизации;
- сложность массового изменения IP-адресов;
- относительная сложность обработки заголовков пакетов IPv4.

В спецификации RFC 1726 представлен набор функций, основными среди них являются:

- масштабируемость: идентификация и определение адресов как минимум 10^{12} конечных систем и 10^{9} индивидуальных сетей;
- топологическая гибкость: архитектура маршрутизации и протокол должны работать в сетях с различной топологией;
- преемственность: обеспечение чёткого плана перехода от существующей версии IPv4;
- независимость от среды передачи: работа среди множества сетей с различными средами передачи данных со скоростями до сотен гигабит в секунду;
- автоматическое конфигурирование хостов и маршрутизаторов;
- безопасность на сетевом уровне;
- мобильность: обеспечение работы с мобильными пользователями, сетями и межсетевыми системами;
- расширяемость: возможность дальнейшего развития в соответствии с новыми потребностями.

В результате реализации заявленных функций важнейшие инновации IPv6 состоят в следующем:

- упрощен стандартный заголовок IP-пакета;
- изменено представление необязательных полей заголовка;
- расширено адресное пространство;
- улучшена поддержка иерархической адресации, агрегирования маршрутов и автоматического конфигурирования адресов;
- введены механизмы аутентификации и шифрования на уровне IP-пакетов;
- введены метки потоков данных.

Header

+	Bits 0–3	4–11	12-15	16–23	24–31
0	Version	Traffic Class	Flow Label		
32	Payload Length			Next Header	Hop Limit
64 96 128 160	Source Address				
192 224 256 288	Destination Address				

The header is in the first 40 octets (320 bits) of the packet and contains:

- Version - version 6 (4-bit IP version).
- Traffic class - packet priority (8 bits). Priority values subdivide into ranges: traffic where the source provides congestion control and non-congestion control traffic.
- Flow label - QoS management (20 bits). Originally created for giving real-time applications special service, but currently unused.
- Payload length - payload length in bytes (16 bits). When cleared to zero, the option is a "Jumbo payload" (hop-by-hop).
- Next header - Specifies the next encapsulated protocol. The values are compatible with those specified for the IPv4 protocol field (8 bits).
- Hop limit - replaces the time to live field of IPv4 (8 bits).
- Source and destination addresses - 128 bits each.

The *protocol* field of IPv4 is replaced with a *next header* field. This field usually specifies the transport layer protocol used by a packet's payload. In the presence of options, however, the *next header* field specifies the presence of one or more out of six *extension headers*, which then follow the IPv6 header in distinct order; the payload's protocol itself is specified in the next header field of the last extension header.

Extension Header	Type	Size	Description	RFC
Hop-By-Hop Options	0	variable	Options that need to be examined by all devices on the path.	RFC 2460

Routing	43	variable	Methods to specify the route for a datagram. (Used with Mobile IPv6)	RFC 2460, RFC 3775, RFC 5095
Fragment	44	64 bits	Contains parameters for fragmentation of datagrams.	RFC 2460
Authentication Header (AH)	51	variable	Contains information used to verify the authenticity of most parts of the packet. (See IPsec)	RFC 4302
Encapsulating Security Payload (ESP)	50	variable	Carries encrypted data for secure communication. (See IPsec).	RFC 4303
Destination Options	60	variable	Options that need to be examined only by the destination of the packet.	RFC 2460
No Next Header	59	empty	A placeholder indicating no next header.	RFC 2460

The payload can have a size of up to 64 KB in standard mode, or larger with a "jumbo payload" option in a *Hop-By-Hop Options* extension header.

Fragmentation is handled only in the sending host in IPv6: routers never fragment a packet, and hosts are expected to use Path MTU discovery.

В протоколе 6 версии (IPv6)

Версия (4 бита)	Класс трафика (8 бит)	Метка потока (20 бит)
Длина полезной нагрузки (16 бит)	След. заголовок (8 бит)	Число переходов
IP-адрес отправителя (128 бит)		
IP-адрес получателя (128 бит)		
Данные		

- Версия — для IPv6 значение поля должно быть равно 6.
- Класс трафика — определяет приоритет трафика (QoS, класс обслуживания).
- Метка потока — уникальное число, одинаковое для однородного потока пакетов.
- Длина полезной нагрузки — длина данных (заголовок IP-пакета не учитывается).
- Следующий заголовок — Определяет следующий инкапсулированный протокол.
- Число переходов — максимальное число роутеров, которые может пройти пакет. При прохождении роутера это значение уменьшается на единицу и по достижению нуля пакет отбрасывается.

Способи обєднання мереж, що функціонують на основі протоколів різних версій.

1.Підтримка двох версій протоколу.

2. **Teredo** — сетевой протокол, предназначенный для передачи IPv6 пакетов через сети IPv4, в частности через устройства, работающие по технологии NAT, путём их инкапсуляции в UDP дейтаграммы

Стандартные средства инкапсуляции IPv6 в IPv4 требуют, чтобы как сервер, так и клиент имели собственный IP-адрес. Однако настоящее время в связи с дефицитом IPv4 адресов, многие организации имеют один глобальный IP-адрес для всей сети, используя технологию NAT. Протокол teredo даёт возможность доступа к IPv6 сетям при такой конфигурации

ICMP

ICMP (англ. Internet Control Message Protocol — межсетевой протокол управляющих сообщений) — сетевой протокол, входящий в стек протоколов TCP/IP. В основном ICMP используется для передачи сообщений об ошибках и других исключительных ситуациях, возникших при передаче данных. Также на ICMP возлагаются некоторые сервисные функции. Только информирует об ошибках а не исправляет их.

Функции:

- Переадресация дейтаграмм
- Формирует и выдает сообщение о недосягаемости сети или конкретного узла
- Сообщение о некорректности параметров передачи
- Формирует и пересылает временные метки
- Запрос на удаленную станцию и получение отклика от нее

2 типа ICMP

- Сообщение управления
- Сообщении об ошибки, которые были при передачи

Формат ICMP-пакета			
Бит	**0—7**	**8—15**	**16—31**
0	Тип	Код	Контрольная сумма
32	Содержание сообщения (зависит от значений полей «Код» и «Тип»)		

Тип – 15 типов сообщений. Задается какое именно

Код – служебное поле и указывает на дополнительную иерархию в рамках сообщения

Контрольная сумма – сумма всего сообщения

Данные – необязательное, но чаще всего там первые 64 бита сообщения, для которого сформировано данное сообщение

Типы ICMP пакетов (полный список)

- **3** — Адресат недоступен

код 0 — Сеть недостижима
код 1 — Хост недостижим
код 2 — Протокол недостижим
код 3 — Порт недостижим
код 4 — Необходима фрагментация, но установлен флаг ее запрета (DF)
код 5 — Неверный маршрут от источника
код 6 — Сеть назначения неизвестна
код 7 — Хост назначения неизвестен
код 8 — Хост источник изолирован
код 9 — Сеть административно запрещена
код 10 — Хост административно запрещен
код 11 — Сеть недоступна для TOS
код 12 — Хост недоступен для TOS
код 13 — Коммуникации административно запрещены

- **11** — Превышение временного интервала (для дейтаграммы время жизни истекло)

код 0 — Время жизни пакета (TTL) истекло при транспортировке
код 1 — Время жизни пакета (время сборки фрагментов) истекло при дефрагментации

- Задание фиксированного маршрута сообщения. В поле данных задаются ip-адресса тех маршрутизаторов, шлюзов, и пр узлов, через которые должно пройти сообщение.

Существует ICMPv6 для IPv6

ARP, RARP

ARP (англ. *Address Resolution Protocol* — протокол разрешения адресов) — сетевой протокол канального уровня, предназначенный для преобразования IP-адресов (адресов сетевого уровня) в MAC-адреса (адреса канального уровня) в сетях TCP/IP

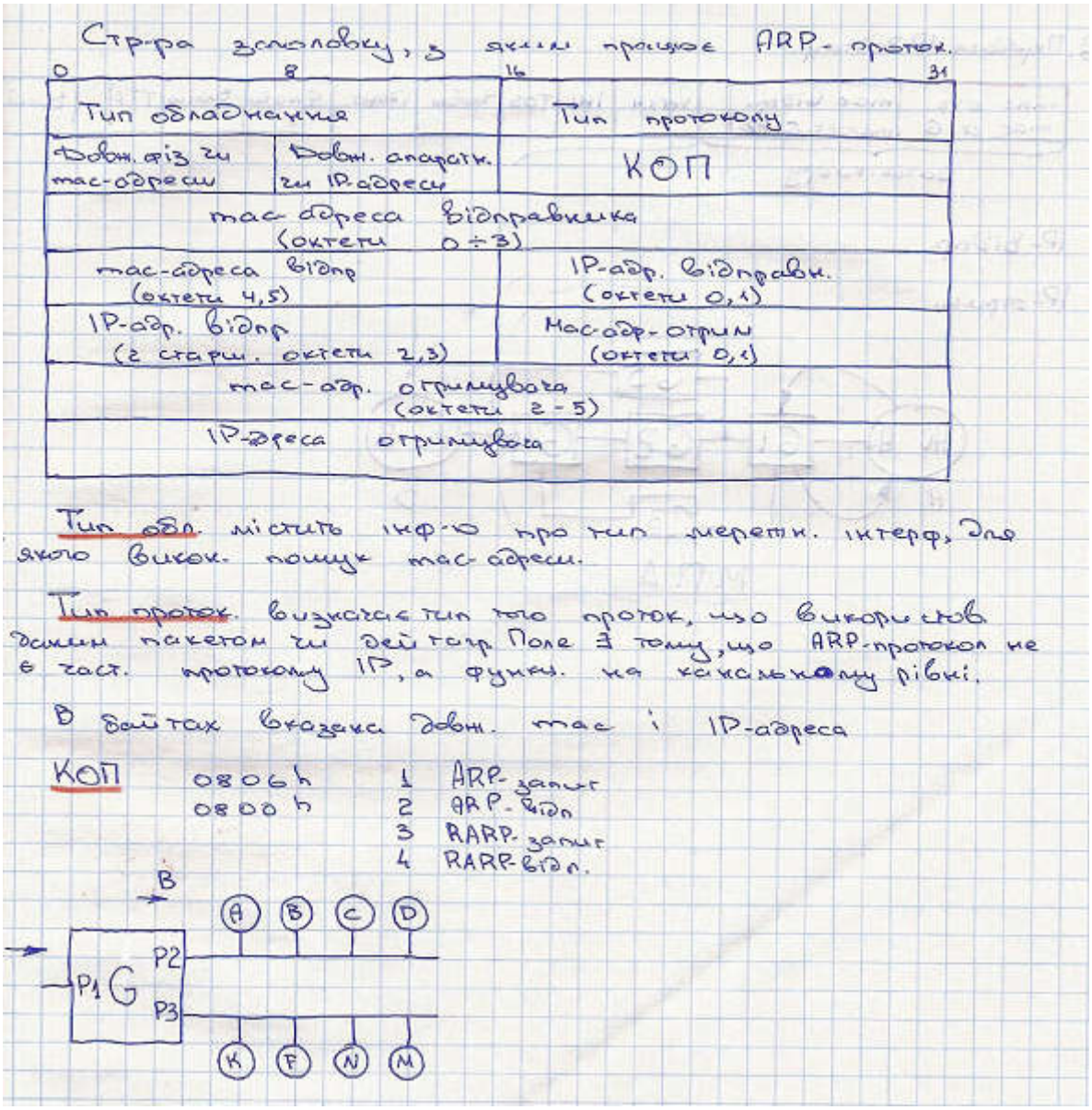
На каждом порту есть **ARP** таблица, в которой содержаться соответствующие мак и айпи адреса, а так же информация о типе сетевого интерфеса и статус записи (статическая или динамическая)

Когда нужно переслать пакет по соответствующему айпи адресу выбирается мак адрес на который и отправляется пакет. Если таких данных нет или они неактивны выполняются следующие действия:

- По сегменту сети рассылается широковещательный **ARP** запрос
- Дейтаграмму, которую нужно передать, записывают в очередь пакетов
- Все станции в сегменте сравнивают свои айпи адреса с адресом в **ARP** запросе
- Станция, которая имеет указанный айпи адрес посылает **ARP** ответ, в котором указывает свой мак адрес
- После получения **ARP** ответа в таблице создается или активизируется соответствующая запись мак и айпи адреса
- Осуществляется передача айпи дейтаграмма в конечную точку

ARP выполняет 2 функции:

- Определяет мак адрес при передаче дейтаграммы или пакета
- Отвечает на **ARP** запросы



Протокол UDP

UDP (англ. *User Datagram Protocol* — протокол пользовательских датаграмм) — это транспортный протокол для передачи данных в сетях IP без установления соединения. Он является одним из самых простых протоколов транспортного уровня модели OSI. Протокол не требует установки соединения и передает сообщении в дейтаграммном режиме. Он не контролирует последовательность передачи фрагментов, не не контролирует дубликаты дейтаграмм, не контролирует ошибки при передаче.

Первые 64 бита (8 октетов) датаграммы представляют собой UDP-заголовок, остальные биты — данные сообщения:

Биты	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0-31	Порт отправителя (Source port)																Порт получателя (Destination port)															
32-63	Длина датаграммы (Length)																Контрольная сумма (Checksum)															
64-...	Данные (Data)																															

Source port - обычно 0 (не нужно подтверждение приема)

Destination port - **********.

Если необходимо подтверждение то Source port не равен 0 (доп сигнал про необходимость квитанции)

Значение поля «длина датаграммы» указывает на длину *всего* UDP-сообщения, то есть включая и UDP-заголовок. Измеряется в октетах.

Контрольная сумма – контролирует корректность всей передачи (заголовок, поле данных, псевдозаголовок)

Псевдозаголовок

UDP-заголовок не содержит информации об адресе отправителя и получателя, поэтому даже при совпадении порта получателя нельзя с точностью сказать, что сообщение пришло в нужное место. Для проверки того, что UDP-сообщение достигло пункта своего назначения, используется дополнительный псевдозаголовок:

Биты	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
0-31	IP-адрес отправителя (Source address)																															
32-63	IP-адрес получателя (Destination address)																															
64-95	0	0	0	0	0	0	0	0	Протокол (Protocol)								Длина UDP-датаграммы (UDP length)															

Поле «протокол» содержит в себе значение 17 (00010001 в двоичном виде, 0x11 — в шестнадцатеричном) — идентификатор UDP-протокола. Поле «длина UDP-датаграммы» содержит в себе длину UDP-сообщения (UDP-заголовок + данные; длина псевдозаголовка не учитывается) в октетах, то есть совпадает с одноименным полем в UDP-заголовке.

Псевдозаголовок не включается в UDP-сообщение. Он используется для расчета контрольной суммы перед отправлением сообщения и при его получении (получатель составляет свой псевдозаголовок, используя адрес хоста, с которого пришло сообщение, и собственный адрес, а затем считает контрольную сумму).

TCP

TCP – это протокол, который выполняет передачу данных пользователя в режиме зеркального соединения.

Выполняет функции:

- Установка логического соединения между системами удаленных станций
- Контроль последовательности передачи пакетов сообщения
- Выявление и, по возможности, исправление ошибок. Возникших при передаче.

Формат TCP-сегмента

Формат TCP-сегмента				
Бит	**0 — 3**	**4 — 9**	**10 — 15**	**16 — 31**
0	Порт источника			Порт назначения
32	Номер последовательности			
64	Номер подтверждения			
96	Смещение данных	Зарезервировано	Флаги	Окно
128	Контрольная сумма			Указатель важности
160	Опции (необязательное)			
160/192+	Данные			

[править] Порт источника

Порт источника идентифицирует порт, с которого отправлены пакеты.

[править] Порт назначения

Порт назначения идентифицирует порт, на который отправлен пакет.

[править] Номер последовательности

Номер последовательности выполняет две задачи:

1. Если установлен флаг SYN, то это начальное значение номера последовательности и первый байт данных — это номер последовательности плюс 1.
2. В противном случае, если SYN не установлен, первый байт данных — номер последовательности

Поскольку TCP-поток в общем случае может быть длинее, чем число различных состояний этого поля, то все операции с номером последовательности должны выполняться по модулю 2^32. Это накладывается практическое ограничение на использование TCP. Если скорость передачи комуникационной системы такова, чтобы в

течение MSL (максимального времени жизни сегмента) произошло переполнение номера последовательности, то в сети может появиться два сегмента с одинаковым номером, относящихся к разным частям потока, и приёмник получит некорректные данные .

[править] Номер подтверждения

Если установлен флаг ACK, то это поле содержит номер последовательности, ожидаемый получателем в следующий раз. Помечает этот сегмент как подтверждение получения.

В это поле заносится первый байт следующего сегмента, который готова принять станция в след цикле

[править] Смещение данных

Это поле определяет размер заголовка пакета TCP в 32-битных словах. Минимальный размер составляет 5 слов, а максимальный — 15, что составляет 20 и 60 байт соответственно. Смещение считается от начала заголовка TCP.

[править] Зарезервировано

Зарезервировано (6 бит) для будущего использования и должны устанавливаться в ноль. Из них два (7-й и 8-й) уже определены:

- **CWR** (Congestion Window Reduced) — Поле «Окно перегрузки уменьшено» — флаг установлен отправителем, чтоб указать, что получен пакет с установленным флагом ECE (RFC 3168)
- **ECE** (ECN-Echo) — Поле «Эхо ECN» — указывает, что данный хост способен на ECN (явное уведомление перегрузки) и для указания отправителю о перегрузках в сети (RFC 3168)

[править] Флаги (управляющие биты)

Это поле содержит 6 битовых флагов:

- **URG** — Поле *"Указатель важности"* задействовано (англ. *Urgent pointer field is significant*). Сообщение с таким флагом передается даже если окно для отправителя будет закрыто, но станция-получатель будет принимать сегменты с этим флагом.
- **ACK** — Поле *"Номер подтверждения"* задействовано (англ. *Acknowledgement field is significant*)
- **PSH** — (англ. *Push function*) инструктирует получателя протолкнуть данные, накопившиеся в приемном буфере, в приложение пользователя
- **RST** — Оборвать соединения, сбросить буфер (очистка буфера) (англ. *Reset the connection*) Установка нового соединения или перезагрузка данного.
- **SYN** — Синхронизация номеров последовательности (англ. *Synchronize sequence numbers*)
- **FIN** (англ. *final*, бит) — флаг, будучи установлен, указывает на завершение соединения (англ. *FIN bit used for connection termination*).

Окно – кол-во байтов, начиная с байта подтверждения, которое станция-получатель может принять

[править] Контрольная сумма

Поле контрольной суммы — это 16-битное дополненение суммы всех 16-битных слов заголовка и текста. Если сегмент содержит нечетное число октетов в заголовке /или тексте, последние октеты дополняются справа 8 нулями для выравнивания по 16-битовой границе. Биты заполнения (0) не передаются в сегменте и служат только для расчета контрольной суммы. При расчете контрольной суммы значение самого поля контрольной суммы принимается равным 0.

[править] Указатель важности

16-битовое значение положительного смещения от порядкового номера в данном сегменте. Это поле указывает порядковый номер октета которым заканчиваются важные (urgent) данные. Поле принимается во внимание только для пакетов с установленным флагом URG.

Опции – содержит доп информацию для передачи и обработки этого сегмента. Обычно это поле вводится для подальшей модификации протокола. Наиболее используемой является опция, которая устанавливает макс размер сегмента, что может передаваться.

Принципи функціонування

Установка соединения

Процесс начала сеанса TCP называется «тройным рукопожатием».

1. Клиент, который намеревается установить соединение, посылает серверу сегмент с номером последовательности и флагом SYN. (Active)
2. Сервер получает сегмент, запоминает номер последовательности и пытается создать сокет (буфера и управляющие структуры памяти) для обслуживания нового клиента. В случае успеха сервер посылает клиенту сегмент с номером последовательности и флагами SYN и ACK, и переходит в состояние SYN-RECEIVED. В случае неудачи сервер посылает клиенту сегмент с флагом RST. (Passive)
3. Если клиент получает сегмент с флагом SYN, то он запоминает номер последовательности и посылает сегмент с флагом ACK, если он одновременно получает и флаг ACK (что обычно и происходит), то он переходит в состояние ESTABLISHED. Если клиент получает сегмент с флагом RST, то он прекращает попытки соединиться.

Если клиент не получает ответа в течение 10 секунд, то он повторяет процесс соединения заново.

Если сервер в состоянии SYN-RECEIVED получает сегмент с флагом ACK, то он переходит в состояние ESTABLISHED. В противном случае после таймаута он закрывает сокет и переходит в состояние CLOSED.

Процесс называется «тройным рукопожатием», поскольку возможен процесс установления соединения с использованием 4 сегментов (SYN в сторону сервера, ACK в сторону клиента, SYN в сторону клиента, ACK в сторону сервера), но для экономии времени используется 3 сегмента.

Active	Passive
1. SYN(seg=x) →	SYN(seg=x)
2. SYN (seg=y, ACK=x+1)	← SYN(seg=y, ACK=x+1)
3. ACK(y+1) →	

Шлюзові протоколи

Шлюзовые протоколы EGP (Exterior Gateway Protocol), GGP (Gateway-to-gateway protocol), IGP (Interior Gateway Protocol) (RIP, HELLO) отвечают за передачу информации о маршрутизации данных и состоянии сети, а также обрабатывают данные для взаимодействия с локальными сетями;

Подробней см http://www.teorver.ru/tcpip-dlya-professionalov/

Ибо описывать каждый протокол – это уйма времени)))))

Протоколи FTP і Telnet

FTP (англ. *File Transfer Protocol* — протокол передачи файлов) — протокол, предназначенный для передачи файлов в компьютерных сетях. FTP позволяет подключаться к серверам FTP, просматривать содержимое каталогов и загружать файлы с сервера или на сервер; кроме того, возможен режим передачи файлов между серверами (см. FXP).

FTP является одним из старейших прикладных протоколов, появившимся задолго до HTTP, в 1971 году. До начала 90-х годов на долю FTP приходилось около половины трафика в сети Интернет[*источник?*]. Он и сегодня широко используется для распространения ПО и доступа к удалённым хостам.

Протокол FTP относится к протоколам прикладного уровня и для передачи данных использует транспортный протокол TCP. Команды и данные, в отличие от большинства других протоколов передаются по разным портам. Порт 20 используется для передачи данных, порт 21 для передачи команд.

Протокол не шифруется, при аутентификации передаёт логин и пароль открытым текстом. Если злоумышленник находится в одном сегменте сети с пользователем FTP, то, используя сниффер, он может перехватить логин и пароль пользователя, или, при наличии специального ПО, получать передаваемые по FTP файлы без авторизации. Чтобы предотвратить перехват трафика, необходимо использовать протокол шифрования данных SSL, который поддерживается многими современными FTP-серверами и некоторыми FTP-клиентами.

Основные команды

- ABOR - Прервать передачу файла
- CDUP - Сменить директорию на вышестоящую.
- CWD - Сменить директорию.
- DELE - Удалить файл (DELE filename).

- HELP - Выводит список команд принимаемых сервером.
- LIST - Возвращает список файлов директории. Список передается через соединение данных (20 порт).
- MDTM - Возвращает время модификации файла.
- MKD - Создать директорию.
- NLST - Возвращает список файлов директории в более кратком формате чем LIST. Список передается через соединение данных (20 порт).
- NOOP - Пустая операция
- PASV - Войти в пассивный режим. Сервер вернет адрес и порт к которому нужно подключиться чтобы забрать данные. Передача начнется при введении следующих команд RETR, LIST и тд.
- PORT - Войти в активный режим. Например PORT 12,34,45,56,78,89. В отличие от пассивного режима для передачи данных сервер сам подключается к клиенту.
- PWD - Возвращает текущую директорию.
- QUIT - Отключиться
- REIN - Реинициализировать подключение
- RETR - Скачать файл. Перед RETR должна быть команда PASV или PORT.
- RMD - Удалить директорию
- RNFR и RNTO - Переименовать файл. RNFR - что переименовывать, RNTO - во что.
- SIZE - Возвращает размер файла
- STOR - Закачать файл. Перед STOR должна быть команда PASV или PORT.
- SYST - Возвращает тип системы(UNIX, WIN, ...)
- TYPE - Установить тип передачи файла(Бинарный, текстовый)

TELNET (англ. ***TEL****ecommunication* ***NET****work*) — сетевой протокол для реализации текстового интерфейса по сети (в современной форме — при помощи транспорта TCP).

Соединение TELNET - это TCP соединение, используемое для передачи данных, с различной управляющей информацией.

Протокол TELNET базируется на трех основных идеях: первая, концепция "Виртуального Сетевого Терминала" (англ. *Network Virtual Terminal*, NVT); вторая, принцип оговоренных опций; третья, симметричный вид терминалов и процессов.

1. Когда устанавливается TELNET соединение, предполагается, что на каждом конце соединения порождается и завершается "Виртуальный Сетевой Терминал" или ВСТ. ВСТ - это воображаемое устройство которое предоставляет стандартное, доступное через сеть, промежуточное представление классического терминала. Это устраняет необходимость в "серверном" и "клиентском" узлах для хранения информации о характеристиках каждого терминала и договоренностей о взаимодействии. Все узлы, как клиентский, так и серверный, отображают свои локальные характеристики устройства с тем, чтобы выступать в сети как ВСТ, и каждый мог принять похожее отображение с другой стороны. ВСТ предназначен для сведения баланса между чрезмерным ограничением и чрезмерными возможностями.
2. Принцип оговоренных опций охватывает тот факт, что многие хосты скорее всего будут хотеть предоставить дополнительные сервисы до или после их доступности в ВСТ, и многие пользователи захотят иметь сложные терминалы с элементами изысканности, вместо минимальных, для получения таких дополнительных сервисов. Независимые от, но структурированные в TELNET протоколе различные "опции" санкционированы и могут быть использованы с "DO, DON'T, WILL,

WON'T" структурой (обсуждается ниже) для того, чтобы позволить пользователю и серверу сходиться в использовании более продуманного (или отличного) набора соглашений для их TELNET соединения. Такие опции включают изменение набора символов, режима эха, и т.д.
Базовая стратегия для налаживания использования опций - это на одной из сторон (или на обоих) инициировать запрос: будет ли определенная опция иметь какой либо эффект. Другая сторона может либо принять, либо отвергнуть запрос. Если запрос принимается, то опция немедленно вступает в силу; если же опция отвергается, то связанный аспект соединения остается как специфицировано для ВСТ. Очевидно, что сторона может всегда отвергать запрос на включение, и никогда не должна отвергать запрос на отключение некоторой опции начиная с момента когда стороны договорились о поддержке ВСТ.
Синтаксис оговоренной опции должен быть таким, чтобы если обе стороны запросят одновременно опцию, то каждый будет рассматривать запрос с другой стороны как положительное подтверждение этой опции.

3. Симметричность синтаксиса согласования может потенциально привести к бесконечному циклу согласования - когда каждая сторона видит входящие команды не подтверждает их, но для подтверждения посылает новый запрос. Для предотвращения таких циклов, необходимо придерживаться следующих правил:
    - Стороны могут запрашивать только изменение статуса опции; т.е. сторона может не посылать "запрос" только для того, чтобы сообщить, что данная опция поддерживается.
    - Если сторона получает что-то, что интерпретируется как запрос переключения в некоторый режим в котором эта сторона уже находится, то на такой запрос не нужно отправлять подтверждения.
    - Всякий раз, когда одна сторона отправляет команду опции на другую сторону, не важно запрашивая или подтверждая, и использование опции должно иметь какой либо эффект на обработку данных, отсылаемых первой стороной второй стороне, то команда опции должна быть вставлена в потоке данных в том месте, с которого желательно, чтобы опция вступила в силу. (Следует отметить, что пройдет некоторое время между передачей запроса и получением подтверждения, и оно может быть отрицательным. Таким образом, хост может буферизовать данные, после отправки запроса опции и до получения ответа с принятием или отвержением опции, для того, чтобы скрыть "период неопределенности" от пользователя.)

Вероятно, что сразу после установки TELNET соединения, запросы опций будут шквалом передаваться в обоих направлениях соединения, из-за того, что каждая сторона будет пытаться получить наилучший сервис от другой стороны. Кроме того, опции могут быть использованы для динамического изменения характеристик соединения с тем, чтобы соответствовать изменяющимся локальным условиям.

Возможно что запросы, инициированные процессами, будут способствовать незаконченному циклу запроса, в случае когда процесс отвечает отказом на повторный запрос опции. Для исключения такой ситуации отклоненные запросы не должны быть повторены, пока что либо не измениться. Операционно, это может означать, что процесс начал выполнять другую программу или пользователь дал другую команду, или что либо случилось в контексте данного процесса или опции. Необходимо принять за правило то, что повторный запрос может быть послан только как результат более поздней информации с другой стороны или в случае локального вмешательства человека.

Проектировщики опции не должны себя чувствовать стесненными несколько ограниченным синтаксисом для оговоренной опции. Смысл простого синтаксиса состоит в том, чтобы упростить принятие опции. Если некоторая специфическая опция требует более сложной структуры согласований чем имеющаяся в "DO, DON'T, WILL, WON'T", то необходимо сначала договориться через существующую структуру согласований, а после того, как обе стороны удостоверятся в понимании этой опции, использовать свободно более экзотический синтаксис.

Протокол TELNET был сделан максимально симметричным по отношению к связке пользователь-сервер, для того, чтобы он легко и естественно покрывал случаи пользователь-пользователь и сервер-сервер. Желательно, но не обязательно, чтобы опции сохраняли этот принцип. В любом случае явно признается, что симметрия - это принцип, а не правило.

Сопутствующий документ "TELNET Option Specifications" предназначен для того, чтобы черпать из него информацию о процедуре написания новых опций.

DNS

DNS (англ. *Domain Name System* — система доменных имён) — распределённая система (распределённая база данных), способная по запросу, содержащему доменное имя хоста (компьютера или другого сетевого устройства), сообщить IP адрес или (в зависимости от запроса) другую информацию. DNS работает в сетях TCP/IP. Как частный случай, DNS может хранить и обрабатывать и обратные запросы, определения имени хоста по его IP адресу - IP адрес по определённому правилу преобразуется в доменное имя, и посылается запрос на информацию типа "PTR".

DNS обладает следующими характеристиками:

- *Распределённость хранения информации*. Каждый узел сети в обязательном порядке должен хранить только те данные, которые входят в его *зону ответственности* и (возможно) адреса *корневых DNS-серверов*.
- *Кеширование информации*. Узел *может* хранить некоторое количество данных не из своей зоны ответственности для уменьшения нагрузки на сеть.
- *Иерархическая структура*, в которой все узлы объединены в дерево, и каждый узел может или самостоятельно определять работу нижестоящих узлов, или *делегировать* (передавать) их другим узлам.
- *Резервирование*. За хранение и обслуживание своих узлов (зон) отвечают (обычно) несколько серверов, разделённые как физически, так и логически, что обеспечивает сохранность данных и продолжение работы даже в случае сбоя одного из узлов.

DNS важна для работы Интернета, ибо для соединения с узлом необходима информация о его IP-адресе, а для людей проще запоминать буквенные (обычно осмысленные) адреса, чем последовательность цифр IP-адреса. В некоторых случаях это позволяет использовать виртуальные серверы, например, HTTP-серверы, различая их по имени запроса. Первоначально преобразование между доменными и IP-адресами производилось с использованием специального текстового файла HOSTS, который составлялся централизованно и обновлялся на каждой из машин сети вручную. С ростом Сети возникла необходимость в эффективном, автоматизированном механизме, которым и стала DNS.

Думаю не нужно, но вдруг кому-то интересно

ключевыми понятиями DNS являются:

- **Зона** — логический узел в дереве имён. Право администрировать зону может быть передано третьим лицам, за счёт чего обеспечивается распределённость базы данных. При этом персона, передавшая право на управление в своей базе данных хранит информацию только о существовании зоны (но не подзон!), информацию о персоне (организации), управляющей зоной, и адрес серверов, которые отвечают за зону. Вся дальнейшая информация хранится уже на серверах, ответственных за зону.
- **Доме́н** — название зоны в системе доменных имён (DNS) Интернета, выделенной какой-либо стране, организации или для иных целей. Структура доменного имени отражает порядок следования зон в иерархическом виде; доменное имя читается слева направо от младших доменов к доменам высшего уровня (в порядке повышения значимости), корневым доменом всей системы является точка ('.'), следом идут домены первого уровня (географические или тематические), затем - домены второго уровня, третьего и т. д. (например, для адреса `ru.wikipedia.org` домен первого уровня — `org`, второго `wikipedia`, третьего `ru`). На практике точку в конце имени часто опускают, но она бывает важна в случаях разделения между относительными доменами и FQDN (англ. *Fully Qualifed Domain Name*, полностью определённое имя домена).
- **Поддомен** — имя подчинённой зоны. (например, `wikipedia.org` — поддомен домена `org`, а `ru.wikipedia.org` — домена `wikipedia.org`). Теоретически такое деление может достигать глубины 127 уровней, а каждая метка может содержать до 63 символов, пока общая длина вместе с точками не достигнет 254 символов. Но на практике регистраторы доменных имён используют более строгие ограничения.
- **DNS-сервер** — специализированное ПО для обслуживания DNS. DNS-сервер может быть ответственным за некоторые зоны и/или может перенаправлять запросы вышестоящим серверам.
- **DNS-клиент** — специализированная библиотека (или программа) для работы с DNS. В ряде случаев DNS-сервер выступает в роли DNS-клиента.
- **ответственность** (англ. *authoritative*) — признак размещения зоны на DNS-сервере. Ответы DNS-сервера могут быть двух типов: *ответственные* (когда сервер заявляет, что сам отвечает за зону) и *неответственные* (англ. *Non-authoritative*), когда сервер обрабатывает запрос, и возвращает ответ других серверов. В некоторых случаях вместо передачи запроса дальше DNS-сервер может вернуть уже известное ему (по запросам ранее) значение (режим кеширования).
- **DNS-запрос** англ. *DNS query* — запрос от клиента (или сервера) серверу. Запрос может быть *рекурсивным* или *нерекурсивным*. Нерекурсивный запрос либо возвращает данные о зоне, которая находится в зоне ответственности DNS-сервера (который получил запрос) или возвращает адреса корневых серверов (точнее, адрес любого сервера, который обладает большим объёмом информации о запрошенной зоне, чем отвечающий сервер). В случае рекурсивного запроса сервер опрашивает серверы (в порядке убывания уровня зон в имени), пока не найдёт ответ или не обнаружит, что домен не существует. На практике поиск начинается с наиболее близких к искомому DNS-серверов, если информация о них есть в кеше и не устарела, сервер может не запрашивать DNS-серверы). Рекурсивные запросы требуют больше ресурсов от сервера (и создают больше трафика), так что обычно принимаются от "известных" владельцу сервера узлов (например, провайдер предоставляет возможность делать рекурсивные запросы только своим клиентам, в корпоративной сети рекурсивные запросы принимаются только из локального сегмента). Нерекурсивные запросы обычно принимаются ото всех узлов сети (и

осмысленный ответ даётся только на запросы о зоне, которая размещена на узле, на DNS-запрос о других зонах обычно возвращаются адреса корневых серверов).
- субдомен - дополнительное доменное имя 3-го уровня в основном домене. Может указывать как на документы корневого каталога, так и на любой подкаталог основного сервера. Например, если у вас есть домен вида mydomain.ru, вы можете создать для него различные поддомены вида mysite1.mydomain.ru, mysite2.mydomain.ru и т. д.

Система DNS содержит иерархию *серверов DNS*. Каждый домен или поддомен поддерживается как минимум одним *авторитетным сервером DNS* (от англ. *authoritative — авторитетный, заслуживающий доверия; в Рунете применительно к DNS и серверам имен часто употребляют и другие варианты перевода: авторизированный, авторитативный*), на котором расположена информация о домене. Иерархия серверов DNS совпадает с иерархией доменов.

Имя и IP-адрес не тождественны — один IP-адрес может иметь множество имён, что позволяет поддерживать на одном компьютере множество веб-сайтов (это называется виртуальный хостинг). Обратное тоже справедливо — одному имени может быть сопоставлено множество IP-адресов: это позволяет создавать балансировку нагрузки.

Для повышения устойчивости системы используется множество серверов, содержащих идентичную информацию, а в протоколе есть средства, позволяющие поддерживать синхронность информации, расположенной на разных серверах. Существует 13 корневых серверов, их адреса практически не изменяются[1].

Протокол DNS использует для работы TCP- или UDP-порт 53 для ответов на запросы. Традиционно запросы и ответы отправляются в виде одной UDP датаграммы. TCP используется для AXFR-запросов.