Тема 8

Стек протоколов TCP/IP

Содержание темы

- Протоколы сетевого уровня, протокол IP.
- Сетевая адресация (IP-адресация).
- Протокол ARP.
- Доменные имена. Система DNS.
- Протокол DHCP.
- Структура заголовка IP-пакета.
- Маршрутизаторы, протоколы RIP, OSPF, EIGRP, BGP.
- Протокол ICMP.
- IPv6.

Содержание темы

- Понятие порта и сокета.
- Протоколы UDP и TCP.
- Методы квитирования.
- Концепция скользящего окна при передаче данных.

Протоколы сетевого уровня

Протоколы сетевого уровня разделяют на:

- **протоколы продвижения данных** (**IP, IPX и пр.**) – непосредственно отвечают за продвижение пакетов от одного узла сети к другому;
- **протоколы маршрутизации** (**RIP, IS-IS, OSPF, EIGRP, BGP и пр.**) – определяют кратчайшие расстояния в сетях и динамически предоставляют маршрутизаторам необходимую информацию для продвижения пакетов в направлении адресата;
- **вспомогательные протоколы** (**ICMP, IGMP и пр.**) – предназначены для выполнения вспомогательных функций, например, информирование об ошибках и других исключительных ситуациях, возникших при продвижении пакетов.

Протоколы сетевого уровня

Протоколы маршрутизации подразделяются:

по области применения:

- **протоколы междоменной маршрутизации – внешние шлюзовые протоколы** (**Exterior Gateway Protocol**, **EGP**) – **BGP**;
- **протоколы внутридоменной маршрутизации – внутренние шлюзовые протоколы** (**Interior Gateway Protocol**, **IGP**) – **RIP**, **OSPF**, **EIGRP**;

в зависимости от алгоритма маршрутизации:

- **дистанционно-векторные** – **RIP**, **BGP**, **EIGRP**;
- **состояния каналов связи** – **IS-IS**, **OSPF**, **NLSP**.

Протоколы сетевого уровня

В **дистанционно-векторных алгоритмах** (**DVA**) каждый маршрутизатор **периодически** и **широковещательно** рассылает по сети вектор, компонентами которого являются расстояния (измеренные в той или иной метрике) от данного маршрутизатора до всех известных ему сетей.

Получив от некоторого соседа вектор расстояний (дистанций) до известных тому сетей, маршрутизатор наращивает компоненты вектора на величину расстояния от себя до данного соседа и дополняет вектор информацией об известных ему самому других сетях, о которых он узнал непосредственно или из аналогичных объявлений других маршрутизаторов.

Обновленное значение вектора маршрутизатор рассылает своим соседям.

Протоколы сетевого уровня

Дистанционно-векторные алгоритмы хорошо работают только в **небольших сетях**.

В больших сетях они периодически засоряют линии связи интенсивным трафиком, к тому же изменения конфигурации не всегда корректно могут отражаться алгоритмом этого типа, так как маршрутизаторы не имеют точного представления о топологии связей в сети, а располагают только косвенной информацией – вектором расстояний.

Протоколы сетевого уровня

Алгоритмы состояния каналов связей (**LSA**) обеспечивают каждый маршрутизатор информацией, достаточной для построения точного графа связей сети.

Все маршрутизаторы работают на основании одного и того же графа, что делает процесс маршрутизации более устойчивым к изменениям конфигурации.

Каждый маршрутизатор использует граф сети для нахождения оптимальных по некоторому критерию маршрутов до каждой из сетей, входящих в составную сеть.

В результате служебный трафик, создаваемый протоколами LSA, **гораздо менее интенсивный**, чем у протоколов DVA.

Протоколы сетевого уровня

У каждого известного стека протоколов есть собственный набор протоколов, соответствующих сетевому уровню модели OSI:

- **OSI** – ES-ES, IS-IS, CONP, CLNP;
- **TCP/IP** – IP, RIP, OSPF, ICMP, IGMP;
- **IPX/SPX** – IPX, RIP, NLSP;
- **AppleTalk** – DDP.

Наиболее распространенными на сегодняшний день являются следующие протоколы: **IP**, **RIP**, **OSPF**, **EIGRP**, **BGP**, **ICMP**, **IGMP**.

Они имеют несколько актуальных версий, например, для поддержки IPv4 и IPv6.

Протокол IP

Основным протоколом сетевого уровня является **межсетевой протокол** (**Internet Protocol**, **IP**).

В его задачу входит **продвижение пакета между сетями** – от одного маршрутизатора к другому до тех пор, пока пакет не попадет в сеть назначения.

Протокол IP – это **дейтаграммный** протокол, работающий без установления соединений по принципу доставки с максимальными усилиями (такой тип сетевого сервиса называют также «ненадежным»).

Протокол IP развертывается не только на **хостах**, но и на **всех маршрутизаторах** (**шлюзах**).

IP-адресация

Важную часть технологии TCP/IP составляют задачи **адресации**, к числу которых относятся:

- **согласованное использование адресов различного типа** – эта задача включает отображение адресов разных типов друг на друга (например сетевого IP-адреса на локальный, доменного имени – на IP-адрес);
- **обеспечение уникальности адресов** – в зависимости от типа адреса требуется обеспечивать однозначность адресации в пределах компьютера, подсети, корпоративной сети или Интернета;
- **конфигурирование сетевых интерфейсов и сетевых приложений**.

IP-адресация

Каждая из перечисленных выше задач имеет достаточно простое решение для сети, число узлов которой не превосходит нескольких десятков.

Ключевым словом, которое характеризует принятый в TCP/IP подход к решению этих проблем, является **масштабируемость.**

Процедуры, предлагаемые TCP/IP для назначения, отображения и конфигурирования адресов, одинаково хорошо работают в сетях разного масштаба.

IP-адресация

Для идентификации сетевых интерфейсов используются три типа адресов:

- **локальные** (**аппаратные**) **адреса** – в большинстве технологий LAN (Ethernet, FDDI, Token Ring) для однозначной адресации интерфейсов используются **MAC-адреса**;
- **сетевые адреса** (**IP-адреса**) – для объединения сетей технологии TCP/IP необходима собственная глобальная система адресации, **не зависящая от способов адресации узлов в отдельных сетях**;
- **символьные** (**доменные**) **имена** – пользователи обычно предпочитают работать с более удобными символьными именами компьютеров и удаленных узлов сети.

IP-адресация

Система IP-адресации должна позволять универсальным и однозначным способом идентифицировать **любой интерфейс составной сети**, поэтому необходима уникальная нумерация всех сетей составной сети, а затем нумерация всех узлов в пределах каждой из этих сетей.

Пара, состоящая из **номера сети** и **номера узла**, отвечает поставленным условиям и может являться **сетевым адресом**, или в терминологии TCP/ IP – **IP-адресом.**

IP-адрес идентифицирует не отдельный узел сети (компьютер или маршрутизатор), а одно сетевое соединение или один сетевой интерфейс.

IP-адресация

Каждый раз, когда пакет направляется адресату через составную сеть, в его заголовке указывается IP-адрес узла назначения.

По номеру сети назначения каждый очередной маршрутизатор находит IP-адрес следующего маршрутизатора и на основании этого IP- адреса его локальный адрес.

Между IP-адресом и локальным адресом узла нет никакой функциональной зависимости, поэтому единственный способ установления соответствия – ведение **таблицы**.

Эту задачу решает **протокол разрешения адресов** (**Address Resolution Protocol**, **ARP**).

IP-адресация

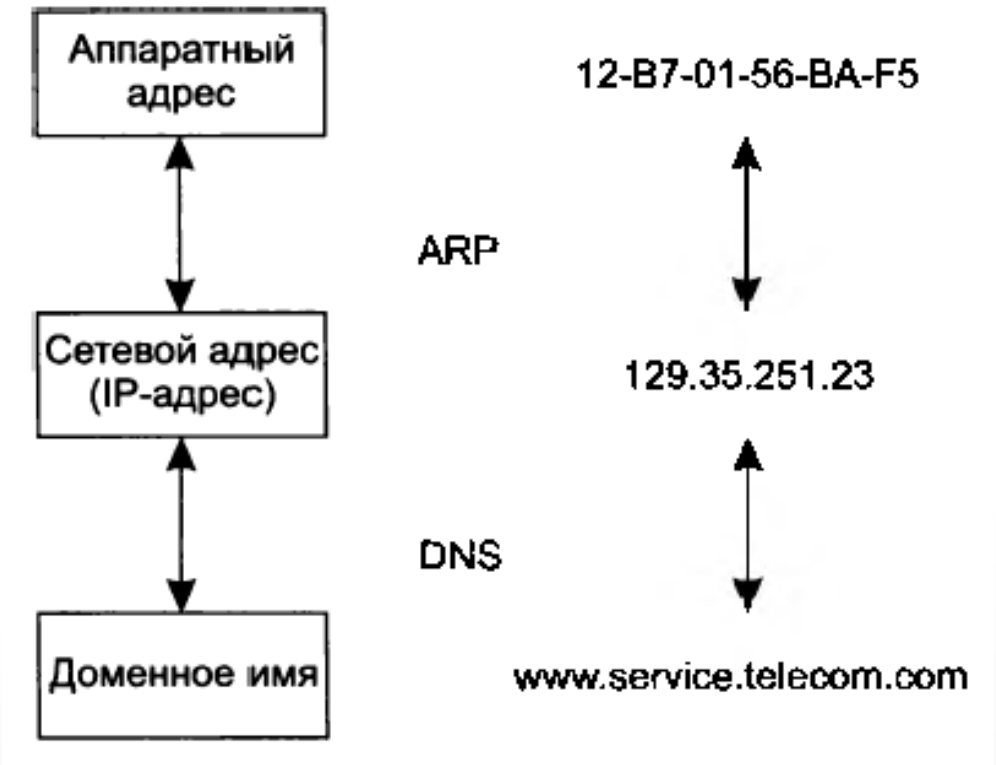
Между IP-адресом узла и его доменным именем (так же, как и локальным адресом) нет никакой функциональной зависимости, поэтому для установления соответствия также требуются таблицы.

В сетях TCP/IP используется специальная сетевая служба, называемая **системой доменных имен** (**Domain Name System**, **DNS)**, которая автоматически устанавливает соответствие между доменными именами и IP-адресами на основании создаваемых администраторами сети таблиц соответствия.

По этой причине доменные имена называют также **DNS-именами.**

IP-адресация

Преобразование IP-адресов.



Формат IP-адреса протокола IPv4

IP-адрес имеет фиксированную длину **4 байта** (**32 бита**) и состоит из двух логических частей – **номера сети** и **номера узла** в сети.

Наиболее распространенной формой представления IP-адреса является запись в виде четырех чисел, представляющих значения каждого байта в десятичной форме и разделенных точками, например:

128.10.2.30

Этот же адрес может быть представлен в двоичном формате:

10000000 00001010 00000010 00011110

А также в шестнадцатеричном формате:

80.0A.02.1D

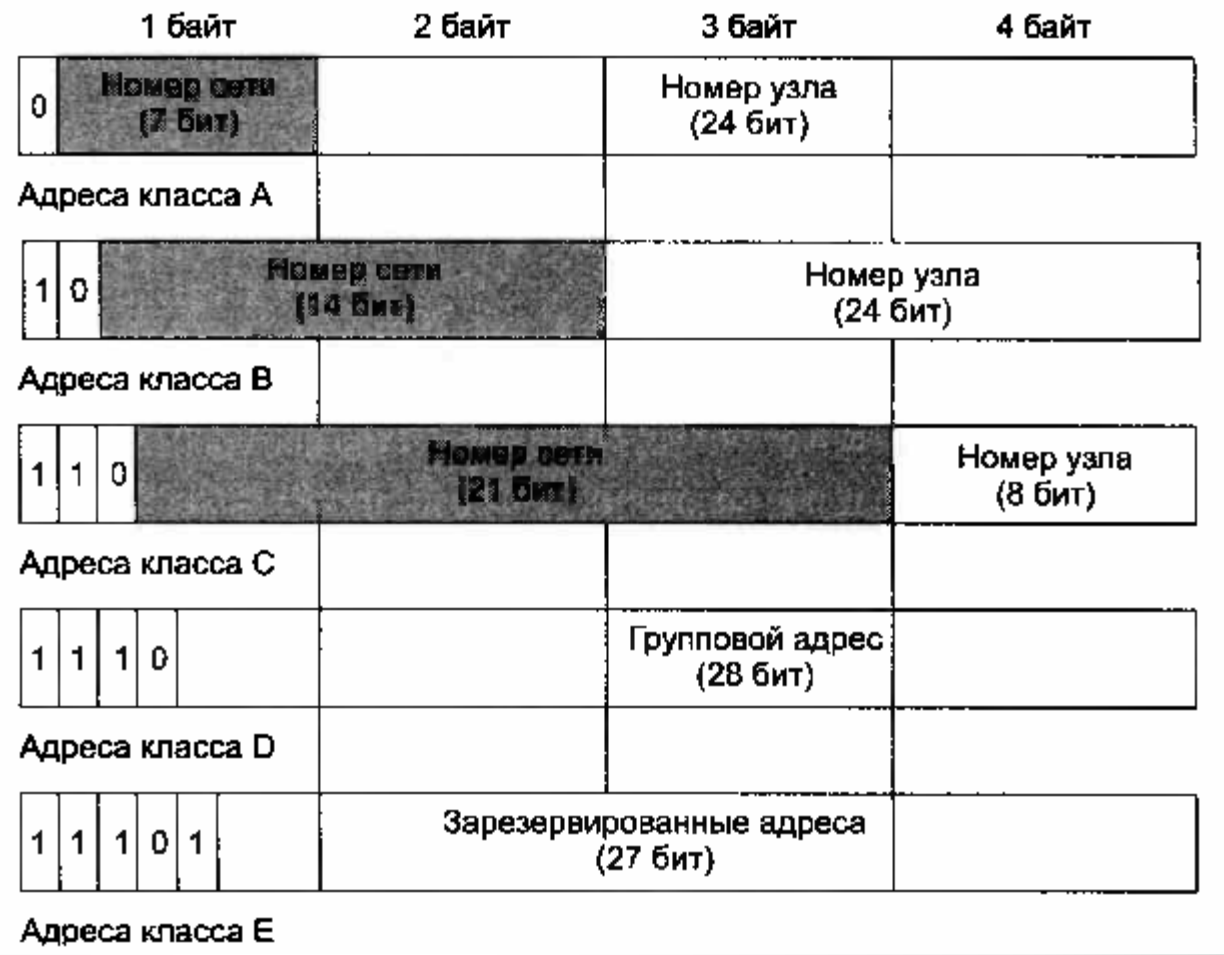
Формат IP-адреса протокола IPv4

Для выделения из IP-адреса IP-адрес сети используют следующие варианты:

- **фиксированные границы** – все 32-битное поле адреса заранее делится на две части (номер сети и номер узла) не обязательно равной, но фиксированной длины;
- **применение маски**, которая позволяет максимально гибко устанавливать границу между номером сети и номером узла – при таком подходе адресное пространство можно использовать для создания множества сетей разного размера;
- **классификация адресов** – компромисс по отношению к двум предыдущим вариантам: размеры сетей хотя и не могут быть произвольными, как при использовании масок, но и не должны быть одинаковыми, как при установлении фиксированных границ.

Классы IP-адресов

Признаком, на основании которого IP-адрес относят к тому или иному классу, являются **значения нескольких первых битов адреса**.



Классы IP-адресов

Диапазоны адресов и максимальное число сетей и узлов, соответствующих каждому классу.

Класс	Первые биты	Наименьший номер сети	Наибольший номер сети	Максимальное число узлов в сети
A	0	1.0.0.0 (0 — не используется)	126.0.0.0 (127 — зарезервирован)	2^{24}, поле 3 байта
B	10	128.0.0.0	191.255.0.0	2^{16}, поле 2 байта
C	110	192.0.0.0	223.255.255.0	2^{8}, поле 1 байт
D	1110	224.0.0.0	239.255.255.255	Групповые адреса
E	11110	240.0.0.0	247.255.255.255	Зарезервировано

Классы IP-адресов

Сетей **класса A** сравнительно немного, зато количество узлов в них очень большое, оно может достигать 2^{24}-2 (16 777 214) узлов.

Сетей **класса B** больше, чем сетей класса A, но их размеры меньше, максимальное количество узлов в сетях класса B составляет 2^{16}-2 (65 534).

Сетей **класса C** больше всего, но они характеризуются самым маленьким максимально возможным количеством узлов, всего – 2^{8}-2 (254).

Классы IP-адресов

Групповые адреса (multicast address) класса D идентифицируют группу сетевых интерфейсов, которые в общем случае могут принадлежать разным сетям.

Интерфейс, входящий в группу, получает наряду с обычным индивидуальным IP-адресом еще один групповой адрес.

Если при отправке пакета в качестве адреса назначения указан адрес класса D, то такой пакет должен быть доставлен всем узлам, которые входят в группу.

Классы IP-адресов

В TCP/IP существуют ограничения при назначении IP-адресов:

- номера сетей и номера узлов **не могут состоять из одних двоичных нулей или единиц**;
- IP-адрес, первый октет которого равен **127**, является **внутренним адресом стека протоколов** компьютера (или маршрутизатора) и **используется для тестирования программ**, а также для организации работы клиентской и серверной частей приложения, установленных на одном компьютере;
- **групповые адреса**, относящиеся к классу D, не делится на номера сети и узла и обрабатывается маршрутизатором особым образом.

Классы IP-адресов

Особенности ограничений IP-адресов:

- если **IP-адрес** состоит **только из двоичных нулей**
  00000000 00000000 00000000 00000000,
  то он называется **неопределенным адресом** и обозначает адрес того узла, который сгенерировал этот пакет;
- если в **поле номера сети** стоят **только нули**
  0.0.x.x (класс B),
  то по умолчанию считается, что узел назначения принадлежит той же самой сети, что и узел, который отправил пакет;

Классы IP-адресов

Особенности ограничений IP-адресов:

- если все **двоичные разряды IP-адреса равны 1**
  11111111 11111111 11111111 11111111,
  то пакет с таким адресом назначения должен рассылаться всем узлам, находящимся в той же сети, что и источник этого пакета – такой адрес называется **ограниченным широковещательным** (**limited broadcast**);
- если в поле адреса назначения в разрядах, соответствующих **номеру узла**, **стоят только единицы**
  x.x.255.255 (класс B),
  то пакет, имеющий такой адрес, рассылается **всем** узлам сети, номер которой указан в адресе назначения – такой тип адреса называется **широковещательным** (**broadcast**).

Классы IP-адресов

В IP-сети **запрещается** присваивать сетевым интерфейсам IP-адреса, начинающиеся со значения 127.

Когда программа посылает данные по IP-адресу 127.x.x.x, то данные возвращаются модулям верхнего уровня того же компьютера как только что принятые.

Такой маршрут перемещения данных образует «**петлю**», поэтому этот адрес называется **адресом обратной петли** (**loopback**).

IP-адреса автономной сети

Когда дело касается сети, являющейся частью Интернета, **уникальность нумерации** может быть обеспечена только **усилиями** специально созданных для этого **центральных органов**.

В небольшой же **автономной IP-сети** условие уникальности номеров сетей и узлов может быть выполнено «**вручную**» **сетевым администратором**.

Однако, **произвольно выбранные адреса данной сети** могут совпасть с **централизовано назначенными адресами Интернета**.

IP-адреса автономной сети

В стандартах Интернета определено несколько диапазонов **частных адресов,** рекомендуемых для автономного использования:

- в классе A – сеть **10.0.0.0**;
- в классе B – диапазон из 16 номеров сетей (**172.16.0.0** – **172.31.0.0**);
- в классе C – диапазон из 255 сетей (**192.168.0.0** – **192.168.255.0**).

Эти адреса **исключены** из множества централизованно распределяемых адресов и **составляют** огромное адресное пространство, достаточное для нумерации узлов автономных сетей практически любых размеров.

Использование масок при адресации

Снабжая каждый IP-адрес **маской**, можно отказаться от понятий классов адресов и сделать систему адресации **более гибкой**.

Виды представления масок для классов адресов:

Класс адресов	Десятичная форма	Двоичная форма	Шестнадцатеричная форма	Префикс
A	255.0.0.0	11111111. 00000000. 00000000. 00000000	FF.00.00.00	/8
B	255.255.0.0	11111111. 11111111. 00000000. 00000000	FF.FF.00.00	/16
C	255.255.255.0	11111111. 11111111. 11111111. 00000000	FF.FF.FF.00	/24

Использование масок при адресации

Например, для IP-адреса **131.128.116.7** указана маска **255.255.192.0**, то есть в двоичном виде IP-адрес 131.128.116.7 равен:

10000011.10000000.01110100.00000111,

в то время как маска 255.255.192.0 выглядит так:

11111111.11111111.11000000.00000000.

Если использовать маску, то 18 последовательных двоичных единиц в маске, «наложенные» на IP-адрес, делят его на две части:

- номер сети: **10000011.10000000.01000000.00000000**;
- номер узла: **00000000.00000000.00110100.00000111**.

Использование масок при адресации

В десятичной форме записи номера сети и узла, дополненные нулями до 32 бит, выглядят соответственно:

- 131.128.64.0;
- 0.0.52.7.

Наложение маски для определения адреса сети можно интерпретировать как выполнение логической операции И (**AND**).

10000011.10000000.01110100.00000111 (IP-адрес)
AND
11111111.11111111.11000000.00000000 (маска)
=
10000011.10000000.01000000.00000000 (адрес сети)

Использование масок при адресации

С помощью масок администратор может разбить одну выделенную поставщиком услуг сеть определенного класса на несколько других, не требуя от него дополнительных номеров сетей, – эта операция называется **разделением на подсети** (**subnetting**).

На основе этого же механизма поставщики услуг могут объединять адресные пространства нескольких сетей путем введения так называемых «префиксов» с целью уменьшения объема таблиц маршрутизации и повышения за счет этого производительности маршрутизаторов – такая операция называется **объединением подсетей** (**supernetting**).

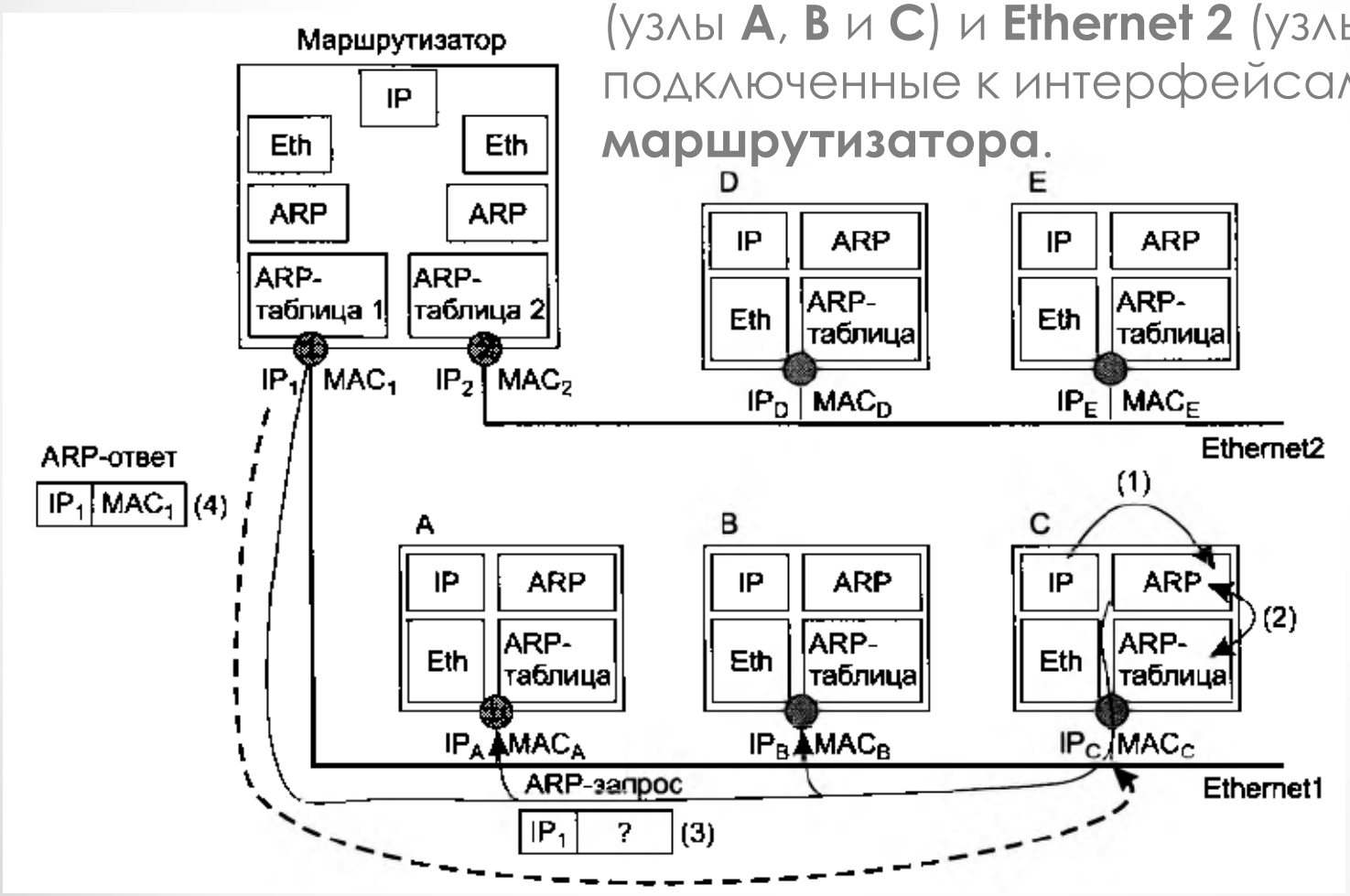
Протокол ARP

Для определения локального адреса по IP-адресу используется **протокол разрешения адресов** (**Address Resolution Protocol**, **ARP**).

Протокол разрешения адресов реализуется различным образом в зависимости от того, работает ли в данной сети протокол локальной сети (**Ethernet**, **Token Ring**, **FDDI**) с возможностью широковещания или же какой-либо из протоколов глобальной сети (**MPLS**, **Frame Relay**, **ATM**), которые, как правило, не поддерживают широковещательный доступ.

Протокол ARP

IP-сеть, включающая две сети – **Ethernet 1** (узлы **A**, **B** и **C**) и **Ethernet 2** (узлы **D** и **E**), подключенные к интерфейсам **1** и **2** **маршрутизатора**.



Узел **C** направляет IP-пакет узлу **D**.

Протокол ARP

1. На первом шаге происходит передача от протокола IP протоколу ARP примерно такого сообщения:
«**Какой MAC-адрес имеет интерфейс с адресом IP_1?**»

2. Работа протокола ARP начинается с просмотра собственной
ARP-таблицы (среди содержащихся в ней записей отсутствует
запрашиваемый IP-адрес).

Протокол ARP

3. В этом случае протокол ARP формирует **ARP-запрос**, вкладывает его в кадр протокола Ethernet и **широковещательно** рассылает.

Зона распространения ARP- запроса ограничивается **сетью Ethernet 1**, так как на пути широковещательных кадров барьером стоит маршрутизатор.

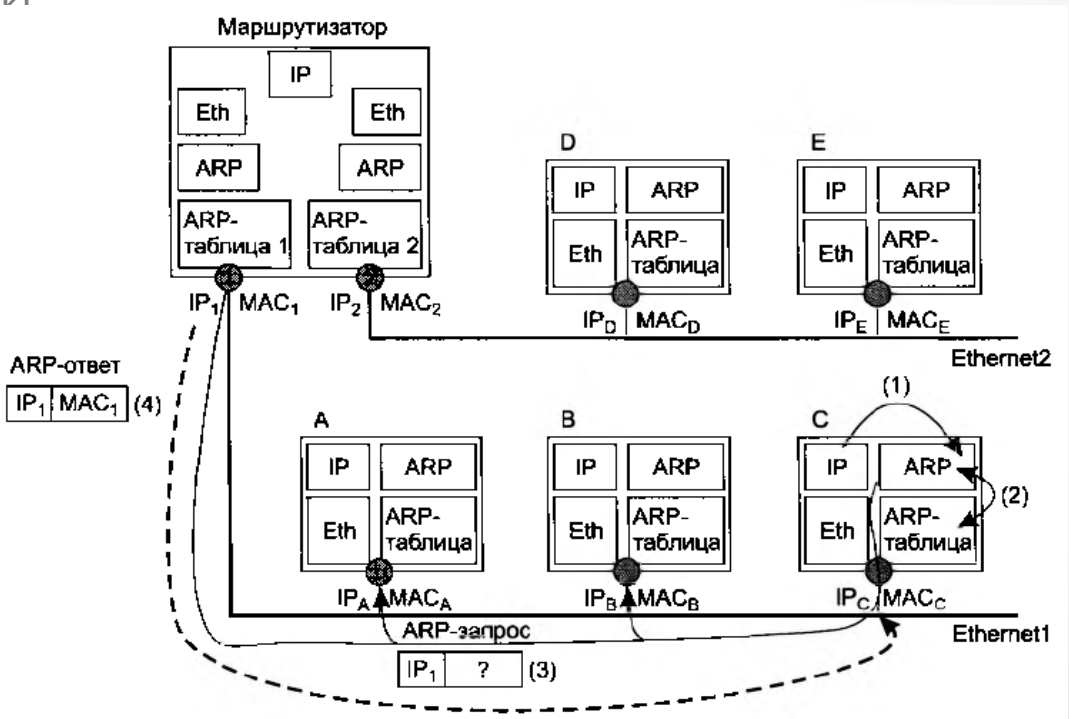
4. Все интерфейсы сети Ethernet 1 получают ARP-запрос и направляют его «своему» протоколу ARP.

ARP сравнивает указанный в запросе адрес IP_1 с IP-адресом собственного интерфейса.

Протокол ARP

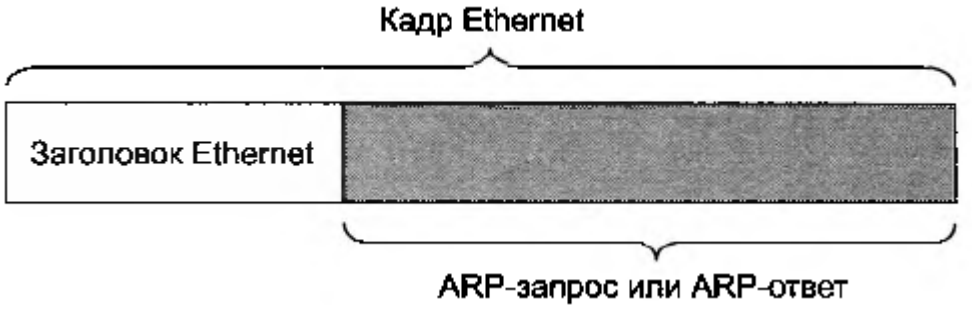
5. Протокол ARP, который констатировал совпадение (**ARP интерфейса 1 маршрутизатора**), формирует ARP-ответ.

В ARP-ответе маршрутизатор указывает **локальный адрес MAC_1** соответствующий адресу IP_1 своего интерфейса, и отправляет его запрашивающему узлу (**C**).



Протокол ARP

ARP-запросы и ARP-ответы имеют один и тот же формат.

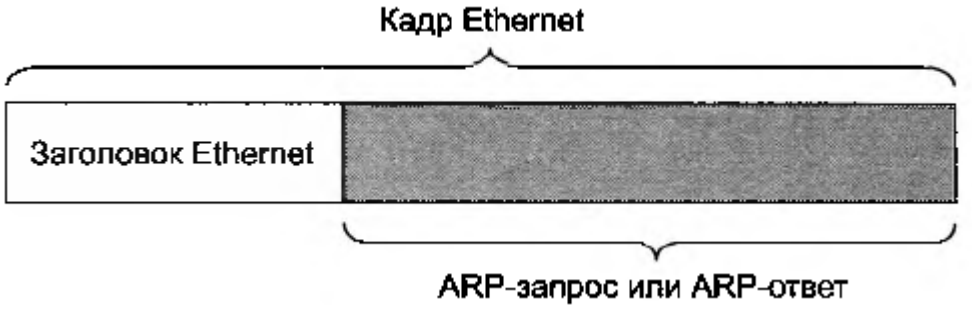


ARP-запрос:

Поле	Значение
Тип сети	1 (0x1)
Тип протокола	2048 (0x800)
Длина локального адреса	6 (0x6)
Длина сетевого адреса	4 (0x4)
Операция	1 (0x1)
Локальный адрес отправителя	008048EB7E60
Сетевой адрес отправителя	194.85.135.75
Локальный (искомый) адрес получателя	000000000000
Сетевой адрес получателя	194.85.135.65

Протокол ARP

ARP-запросы и ARP-ответы имеют один и тот же формат.

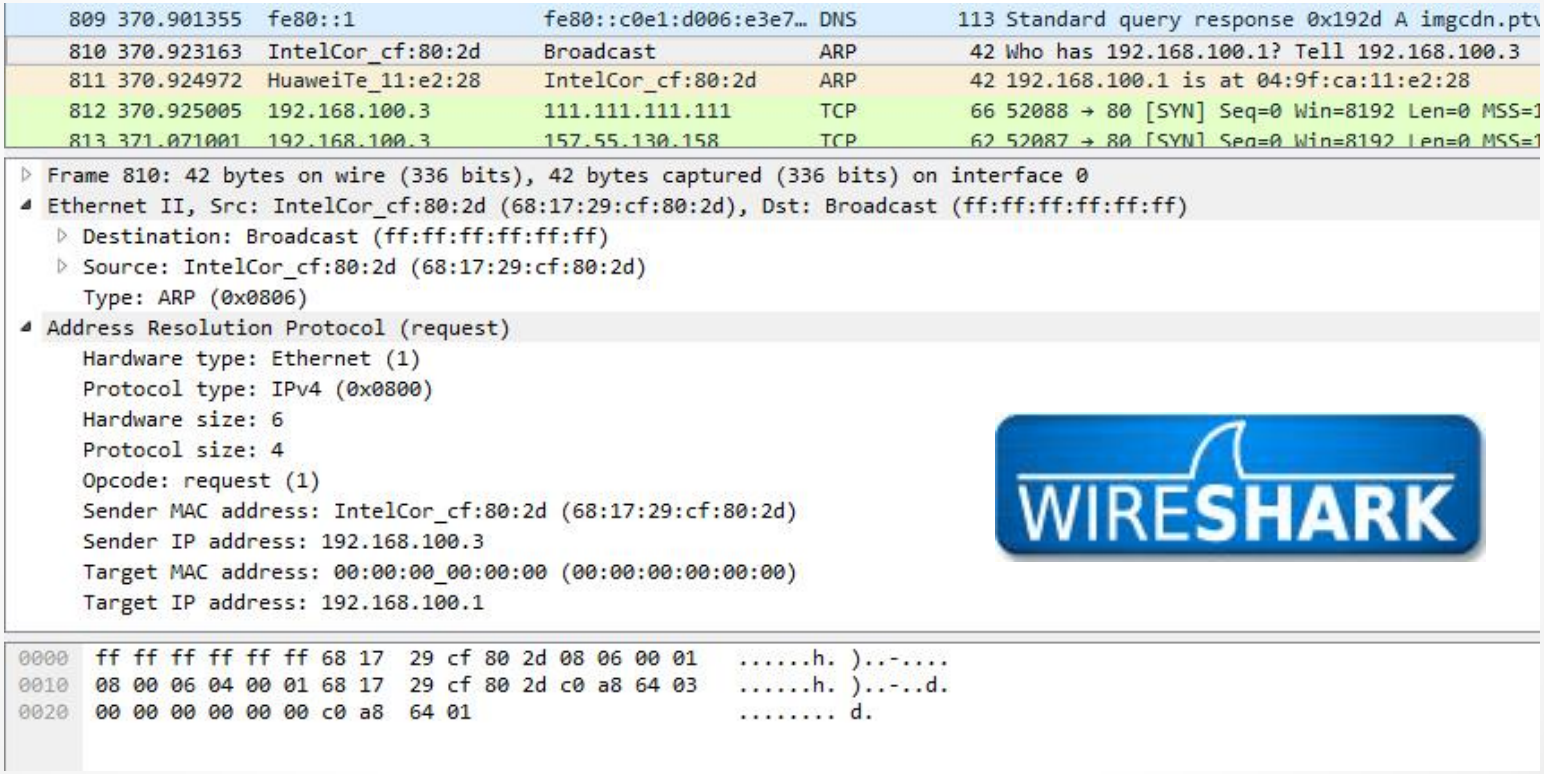


ARP-ответ:

Поле	Значение
Тип сети	1 (0x1)
Тип протокола	2048 (0x800)
Длина локального адреса	6 (0x6)
Длина сетевого адреса	4 (0x4)
Операция	2 (0x1)
Локальный адрес отправителя	00E0F77F1920
Сетевой адрес отправителя	194.85.135.65
Локальный (искомый) адрес получателя	008048EB7E60
Сетевой адрес получателя	194.85.135.75

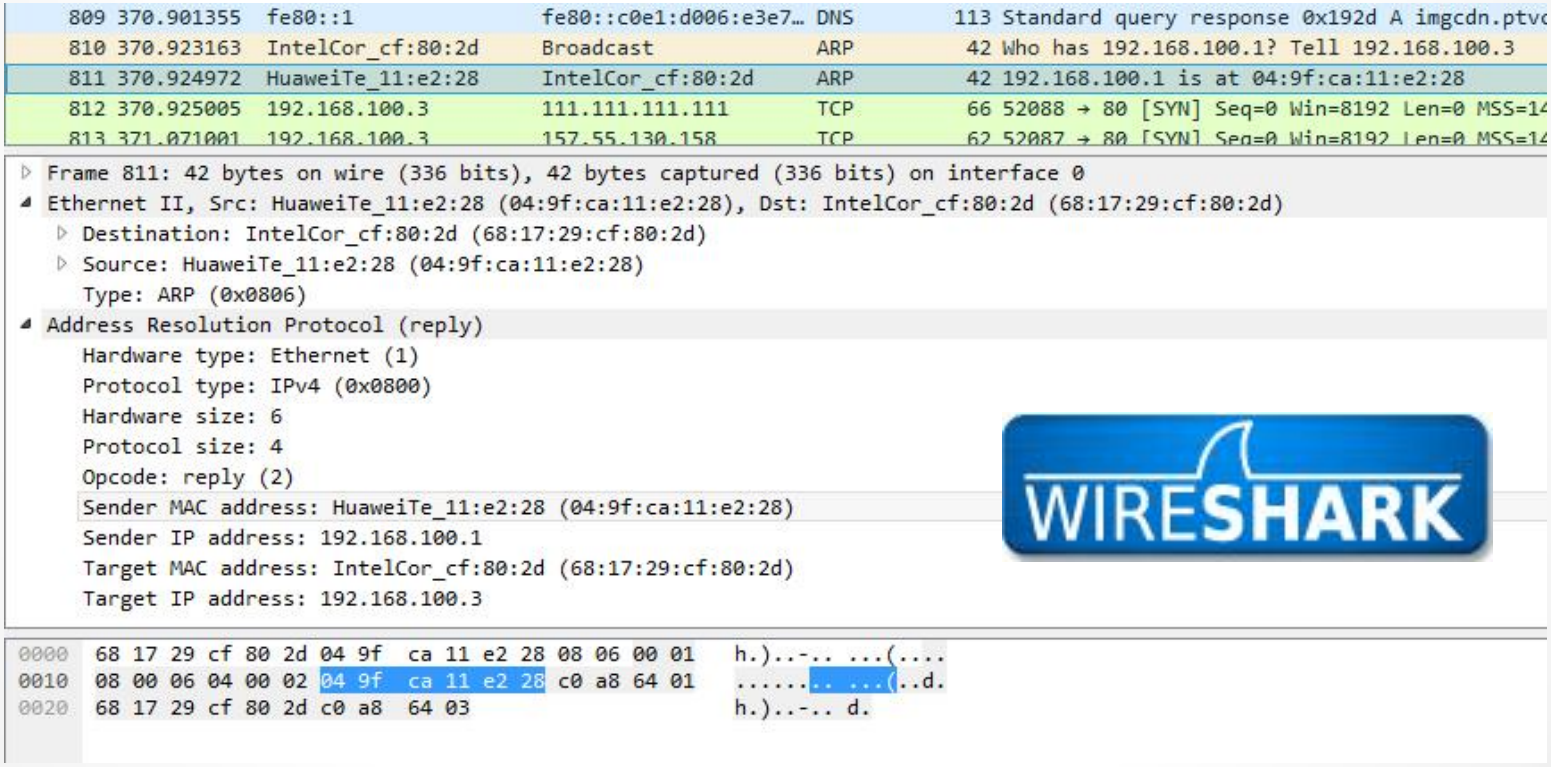
Протокол ARP

ARP-запрос:



Протокол ARP

ARP-ответ:



Протокол ARP

Чтобы уменьшить число ARP-обращений в сети, найденное соответствие между IP-адресом и MAC-адресом сохраняется в **ARP-таблице** соответствующего интерфейса.

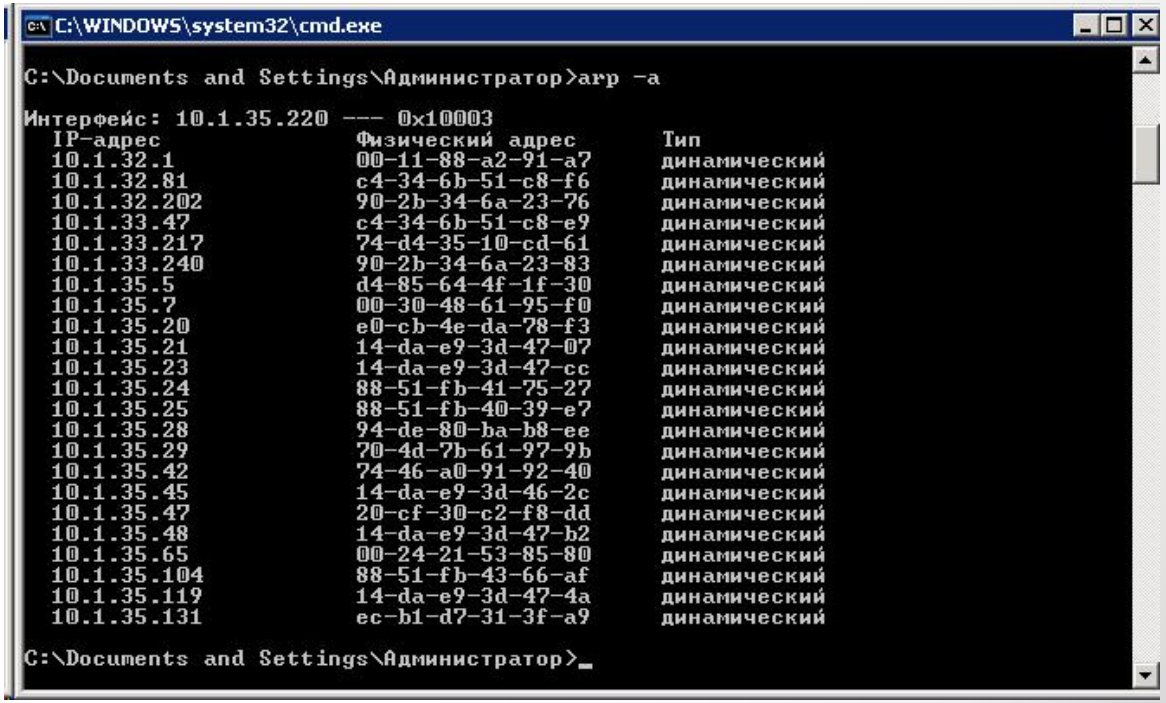
Запись в ARP-таблице появляется **автоматически**, спустя несколько миллисекунд после того, как модуль ARP проанализирует ARP-ответ.

ARP-таблица пополняется не только за счет поступающих на данный интерфейс ARP- ответов, но и в результате извлечения полезной информации из **широковещательных ARP-запросов**.

Протокол ARP

В ARP-таблицах существуют два типа записей:

- **статические записи** создаются вручную с помощью утилиты arp и не имеют срока устаревания, точнее, они существуют до тех пор, пока компьютер или маршрутизатор остается включенным;
- **динамические записи** должны периодически обновляться (если запись не обновлялась в течение нескольких минут, то она исключается из таблицы).



Доменные имена

В стеке TCP/IP применяется **доменная система имен**, которая имеет **иерархическую древовидную структуру**, допускающую наличие в имени произвольного количества составных частей.

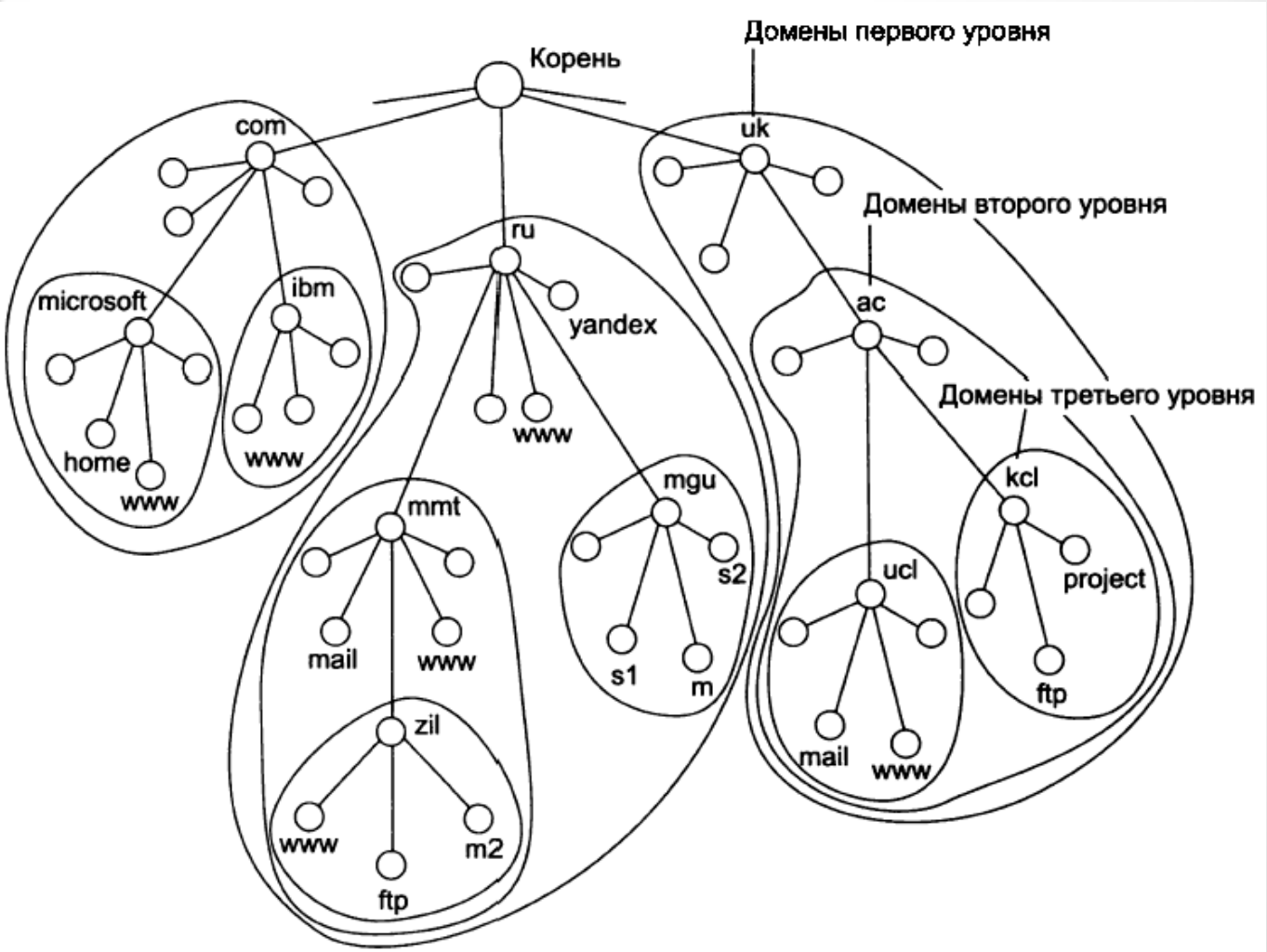
Запись доменного имени **начинается с самой младшей составляющей**, а **заканчивается самой старшей**.

Составные части доменного имени **отделяются друг от друга точкой**.

Разделение имени на части позволяет **разделить административную ответственность** за назначение уникальных имен между различными людьми или организациями в пределах своего уровня иерархии.

Доменные имена

Часть пространства доменных имен:



Доменные имена

Совокупность имен, у которых несколько старших составных частей совпадают, образует **домен имен** (**domain**).

Например, имена **www.zil.mmt.ru**, **ftp.zil.mmt.ru**, **yandex.ru** и **sl.mgu.ru** входят в домен **ru**, так как все они имеют одну общую старшую часть – имя **ru**.

Администратор домена **mgu.ru** несет ответственность за уникальность имен следующего уровня, входящих в домен, то есть имен **s1**, **s2** и **m.**

Образованные домены **s1.mgu.ru**, **s2.mgu.ru** и **rn.mgu.ru** являются **поддоменами** домена **mgu.ru**, так как имеют общую старшую часть имени.

Доменные имена

В доменной системе имен различают:

- **краткое доменное имя** – это имя конечного узла сети (хоста или порта маршрутизатора);
- **относительное доменное имя** – это составное имя, начинающееся с некоторого уровня иерархии, но не самого верхнего (**www.zil**);
- **полное доменное имя** включает составляющие всех уровней иерархии, начиная от краткого имени и кончая корневой точкой (**www.zil.mmt.ru**).

Доменные имена

Корневой домен управляется центральными органами Интернета, в частности организацией **ICANN**.

Домены верхнего уровня назначаются для **каждой страны**, а также для **различных типов организаций**.

Для обозначения стран используются трехбуквенные и двухбуквенные аббревиатуры (**ru** – Россия, **uk** – Великобритания, **us** – Соединенные Штаты).

Для различных типов организаций (**com** – коммерческие организации, **edu** – образовательные организации, **gov** – правительственные организации, **org** – некоммерческие организации, **net** – сетевые организации.

Каждый домен администрирует отдельная организация.

Система DNS

Альтернативой широковещательной рассылке (**ARP**) является применение **централизованной службы**, поддерживающей соответствие между символьными именами и IP-адресами всех компьютеров сети.

На раннем этапе развития Интернета на каждом хосте вручную создавался текстовый **файл отображений** с известным именем **hosts.txt.**

По мере роста Интернета появилась централизованная служба **DNS** (**Domain Name System – система доменных имен**), основанная на распределенной базе отображений «доменное имя – IP-адрес».

Система DNS

Служба DNS имеет **иерархическую структуру**.

Иерархию образуют **DNS-серверы,** которые поддерживают распределенную базу отображений, а **DNS-клиенты** обращаются к серверам с запросами об отображении (разрешении) доменного имени на IP-адрес.

DNS-клиентом является практически каждый узел Интернета (клиентский компьютер, сервер приложений, маршрутизатор).

Запросы к DNS-серверам и их ответы обслуживаются **протоколом DNS**.

Система DNS

Служба DNS использует **текстовые файлы** почти такого же формата, как и файл hosts.txt, состоящие из записей отображений и некоторых служебных записей.

Вершину иерархии серверов DNS составляют **корневые серверы,** они хранят файлы отображений DNS-серверов следующего уровня, называемого **верхним** (**top level DNS**).

Серверы верхнего уровня хранят данные об именах и адресах имен, входящих в домены верхнего уровня, таких как **com**, **ru** или **fm**, а также об именах DNS-серверов, которые обслуживают домены следующего уровня иерархии – второго.

Система DNS

Существуют две основные схемы разрешения DNS-имен:

- реализуется **нерекурсивная** процедура – работу по поиску IP-адреса координирует DNS-клиент;
- реализуется **рекурсивная** процедура.

Система DNS

При реализации **нерекурсивной** процедуры:

- DNS-клиент обращается к корневому DNS-серверу с указанием полного доменного имени;
- DNS-сервер отвечает клиенту, указывая адрес следующего DNS-сервера, обслуживающего домен верхнего уровня, заданный в следующей старшей части запрошенного имени;
- DNS-клиент делает запрос следующего DNS-сервера, который отсылает его к DNS- серверу нужного поддомена, и т. д., пока не будет найден DNS-сервер, в котором хранится отображение запрошенного имени на IP-адрес.

Система DNS

При реализации **рекурсивной** процедуры:

- DNS-клиент запрашивает локальный DNS-сервер, то есть тот сервер, обслуживающий поддомен, которому принадлежит имя клиента;
- далее возможны два варианта действий:
  - если локальный DNS-сервер знает ответ (запрошенное имя входит в тот же поддомен, что и имя клиента, или сервер уже узнавал данное соответствие для другого клиента и сохранил его в своем кэше), то он сразу же возвращает его клиенту;
  - если локальный сервер не знает ответ, то он выполняет итеративные запросы к корневому серверу и т. д. (нерекурсивный вариант), а получив ответ, передает его клиенту, который все это время просто ждет его от своего локального DNS-сервера.

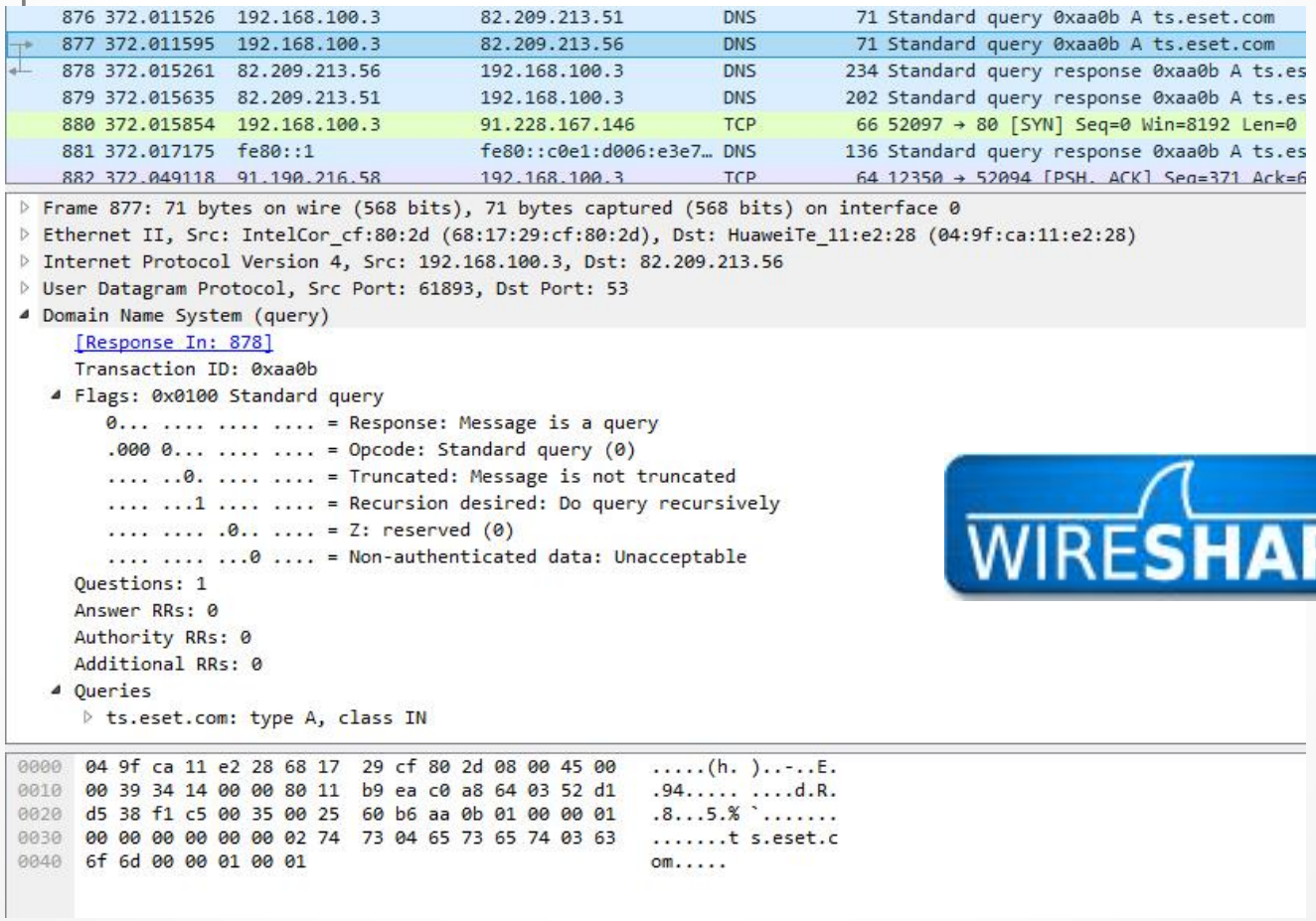
Система DNS

Практически все **резольверы** (программное обеспечение ОС – клиента службы DNS) используют или запрашивают в качестве приоритетной рекурсивную процедуру.

DNS-серверы стараются не поддерживать рекурсивный режим ответов, так как это перегружает их – корневые серверы и серверы верхнего уровня всегда дают нерекурсивные ответы, отсылая серверы нижних уровней к серверам промежуточных уровней.

Система DNS

DNS-запрос:



```
876 372.011526  192.168.100.3    82.209.213.51     DNS   71 Standard query 0xaa0b A ts.eset.com
877 372.011595  192.168.100.3    82.209.213.56     DNS   71 Standard query 0xaa0b A ts.eset.com
878 372.015261  82.209.213.56    192.168.100.3     DNS   234 Standard query response 0xaa0b A ts.es
879 372.015635  82.209.213.51    192.168.100.3     DNS   202 Standard query response 0xaa0b A ts.es
880 372.015854  192.168.100.3    91.228.167.146    TCP   66 52097 → 80 [SYN] Seq=0 Win=8192 Len=0
881 372.017175  fe80::1          fe80::c0e1:d006:e3e7… DNS   136 Standard query response 0xaa0b A ts.es
882 372.049118  91.190.216.58    192.168.100.3     TCP   64 12350 → 52094 [PSH, ACK] Seq=371 Ack=6

▷ Frame 877: 71 bytes on wire (568 bits), 71 bytes captured (568 bits) on interface 0
▷ Ethernet II, Src: IntelCor_cf:80:2d (68:17:29:cf:80:2d), Dst: HuaweiTe_11:e2:28 (04:9f:ca:11:e2:28)
▷ Internet Protocol Version 4, Src: 192.168.100.3, Dst: 82.209.213.56
▷ User Datagram Protocol, Src Port: 61893, Dst Port: 53
◢ Domain Name System (query)
    [Response In: 878]
    Transaction ID: 0xaa0b
  ◢ Flags: 0x0100 Standard query
      0... .... .... .... = Response: Message is a query
      .000 0... .... .... = Opcode: Standard query (0)
      .... ..0. .... .... = Truncated: Message is not truncated
      .... ...1 .... .... = Recursion desired: Do query recursively
      .... .... .0.. .... = Z: reserved (0)
      .... .... ...0 .... = Non-authenticated data: Unacceptable
    Questions: 1
    Answer RRs: 0
    Authority RRs: 0
    Additional RRs: 0
  ◢ Queries
    ▷ ts.eset.com: type A, class IN

0000  04 9f ca 11 e2 28 68 17  29 cf 80 2d 08 00 45 00   .....(h. )..-..E.
0010  00 39 34 14 00 00 80 11  b9 ea c0 a8 64 03 52 d1   .94..... ....d.R.
0020  d5 38 f1 c5 00 35 00 25  60 b6 aa 0b 01 00 00 01   .8...5.% `.......
0030  00 00 00 00 00 00 02 74  73 04 65 73 65 74 03 63   .......t s.eset.c
0040  6f 6d 00 00 01 00 01                              om.....
```

Система DNS

DNS-ответ:

```
877 372.011595  192.168.100.3   82.209.213.56    DNS   71 Standard query 0xaa0b A ts.eset.com
878 372.015261  82.209.213.56   192.168.100.3    DNS   234 Standard query response 0xaa0b A ts.eset.com CN
879 372.015635  82.209.213.51   192.168.100.3    DNS   202 Standard query response 0xaa0b A ts.eset.com CN
880 372.015854  192.168.100.3   91.228.167.146   TCP   66 52097 → 80 [SYN] Seq=0 Win=8192 Len=0 MSS=1460
881 372.017175  fe80::1         fe80::c0e1:d006:e3e7… DNS  136 Standard query response 0xaa0b A ts.eset.com CN
```

```
[Request In: 877]
[Time: 0.003666000 seconds]
Transaction ID: 0xaa0b
Flags: 0x8180 Standard query response, No error
   1... .... .... .... = Response: Message is a response
   .000 0... .... .... = Opcode: Standard query (0)
   .... .0.. .... .... = Authoritative: Server is not an authority for domain
   .... ..0. .... .... = Truncated: Message is not truncated
   .... ...1 .... .... = Recursion desired: Do query recursively
   .... .... 1... .... = Recursion available: Server can do recursive queries
   .... .... .0.. .... = Z: reserved (0)
   .... .... ..0. .... = Answer authenticated: Answer/authority portion was not authenticated by the server
   .... .... ...0 .... = Non-authenticated data: Unacceptable
   .... .... .... 0000 = Reply code: No error (0)
Questions: 1
Answer RRs: 2
Authority RRs: 3
Additional RRs: 3
Queries
   ts.eset.com: type A, class IN
Answers
   ts.eset.com: type CNAME, class IN, cname ts.wip.eset.com
   ts.wip.eset.com: type A, class IN, addr 91.228.167.146
Authoritative nameservers
Additional records
```

```
0000  68 17 29 cf 80 2d 04 9f  ca 11 e2 28 08 00 45 00   h.)..-.. ...(..E.
0010  00 dc f9 a9 00 00 3b 11  38 b2 52 d1 d5 38 c0 a8   ......;. 8.R..8..
0020  64 03 00 35 f1 c5 00 c8  06 03 aa 0b 81 80 00 01   d..5.... ........
0030  00 02 00 03 00 03 02 74  73 04 65 73 65 74 03 63   .......t s.eset.c
0040  6f 6d 00 00 01 00 01 c0  0c 00 05 00 01 00 03 7c   om...... .......|
0050  4f 00 09 02 74 73 03 77  69 70 c0 0f c0 29 00 01   O...ts.w ip...)..
```

Система DNS

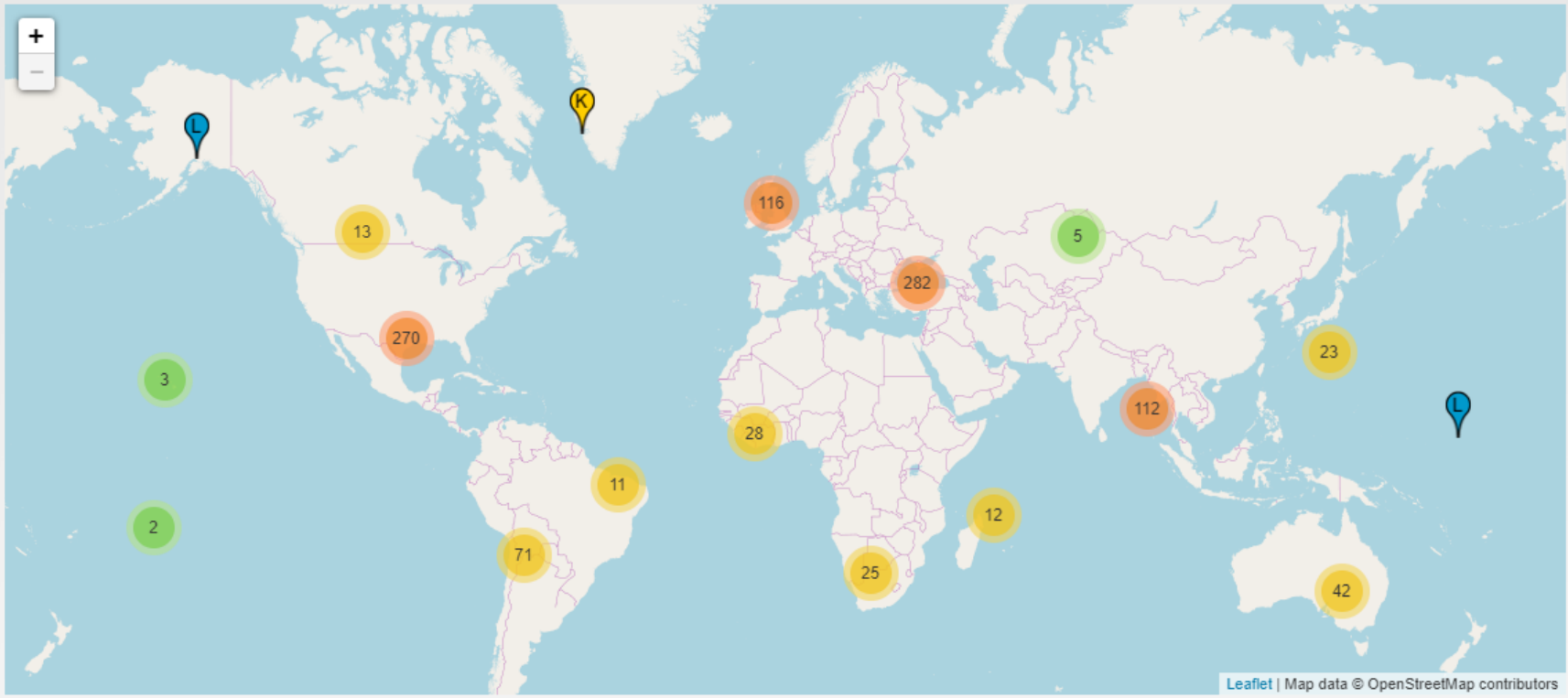
Корневые серверы – наиболее уязвимое звено службы DNS, так как разрешение всех запросов, ответы на которые не находятся в кэше или файле зоны какого-либо DNS-сервера нижнего уровня, начинаются с обращения к одному из корневых серверов.

Изначально было решено обеспечить высокую степень резервирования: было установлено **13** корневых серверов с именами от **a.root-servers.net** до **m.root-servers.net** и 13 IP-адресов.

Все существующие корневые серверы по-прежнему разделяют те же 13 имен и 13 IP-адресов.

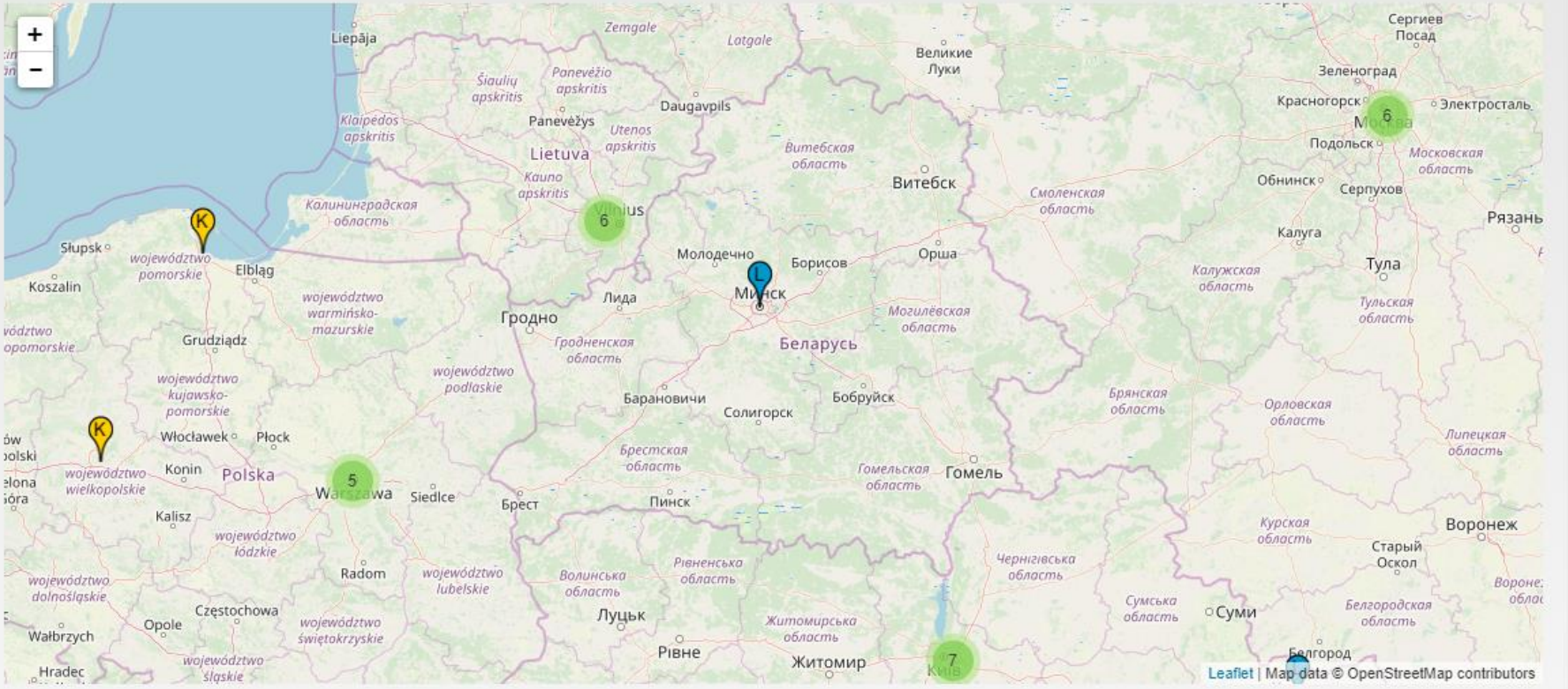
Система DNS

По данным **root-servers.org** на 9 октября 2019 года с учетом зеркал функционирует **1018** корневых сервера.



As of 2019-10-09, the root server system consists of 1018 instances operated by the 12 independent root server operators.

Система DNS


Liepāja
Zemgale
Latgale
Šiaulių apskritis
Panevėžio apskritis
Panevėžys
Daugavpils
Klaipėdos apskritis
Utenos apskritis
Lietuva
Kauno apskritis
Калининградская область
Slupsk
województwo pomorskie
Elbląg
Koszalin
województwo warmińsko-mazurskie
Grudziądz
województwo kujawsko-pomorskie
Włocławek
Płock
województwo wielkopolskie
Konin
Polska
Kalisz
województwo łódzkie
Radom
województwo dolnośląskie
Częstochowa
Opole
Wałbrzych
województwo świętokrzyskie
Hradec
województwo śląskie
Siedlce
województwo podlaskie
województwo lubelskie
Брест
Гродно
Гродненская область
Лида
Молодечно
Борисов
Минск
Беларусь
Могилёвская область
Витебская область
Витебск
Орша
Смоленская область
Великие Луки
Барановичи
Солигорск
Бобруйск
Брестская область
Пинск
Волинська область
Рівненська область
Луцьк
Рівне
Житомирська область
Житомир
Гомельская область
Гомель
Чернігівська область
Сумська область
Суми
Курская область
Воронеж
Старый Оскол
Белгородская область
Белгород
Сергиев Посад
Зеленоград
Красногорск
Электросталь
Подольск
Московская область
Обнинск
Серпухов
Калуга
Рязань
Тула
Калужская область
Тульская область
Брянская область
Орловская область
Липецкая область
Leaflet | Map data © OpenStreetMap contributors

Протокол DHCP

Для нормальной работы сети каждому сетевому интерфейсу компьютера и маршрутизатора должен быть назначен IP-адрес.

Процедура присвоения адресов происходит в ходе **конфигурирования** компьютеров и маршрутизаторов.

Назначение IP-адресов может происходить **вручную**, при этом администратор должен помнить, какие адреса из имеющегося множества он уже использовал для других интерфейсов, а какие еще свободны, кроме того он должен указать дополнительные параметры (маску и IP-адреса шлюза по умолчанию и DNS-сервера и т. п.).

Эта работа представляет для администратора **утомительную процедуру**.

Протокол DHCP

Протокол динамического конфигурирования хостов (**Dynamic Host Configuration Protocol**, **DHCP**) автоматизирует процесс конфигурирования сетевых интерфейсов, гарантируя от дублирования адресов за счет централизованного управления их распределением.

Протокол DHCP работает в соответствии с моделью **клиент-сервер**:

- во время старта системы **компьютер**, являющийся DHCP-клиентом, **посылает в сеть широковещательный запрос** на получение IP-адреса;
- **DHCP-сервер** откликается и **посылает сообщение-ответ**, содержащее IP-адрес и некоторые другие конфигурационные параметры.

Протокол DHCP

DHCP-сервер может работать в разных режимах:

- **ручное назначение статических адресов** – администратор помимо пула доступных адресов снабжает DHCP-сервер информацией о жестком соответствии IP-адресов физическим адресам или другим идентификаторам клиентских узлов;
- **автоматическое назначение статических адресов** – DHCP-сервер самостоятельно, без вмешательства администратора, произвольным образом выбирает клиенту IP-адрес из пула наличных IP-адресов;
- **автоматическое распределение динамических адресов** – DHCP-сервер выдает адрес клиенту на ограниченное время, называемое **сроком аренды.**

Протокол DHCP

Администратор управляет процессом конфигурирования сети, определяя два основных конфигурационных параметра DHCP-сервера:

- **пул адресов**, доступных распределению;
- **срок аренды**.

DHCP-сервер должен находиться в **одной подсети с клиентами**, учитывая, что клиенты посылают ему широковещательные запросы.

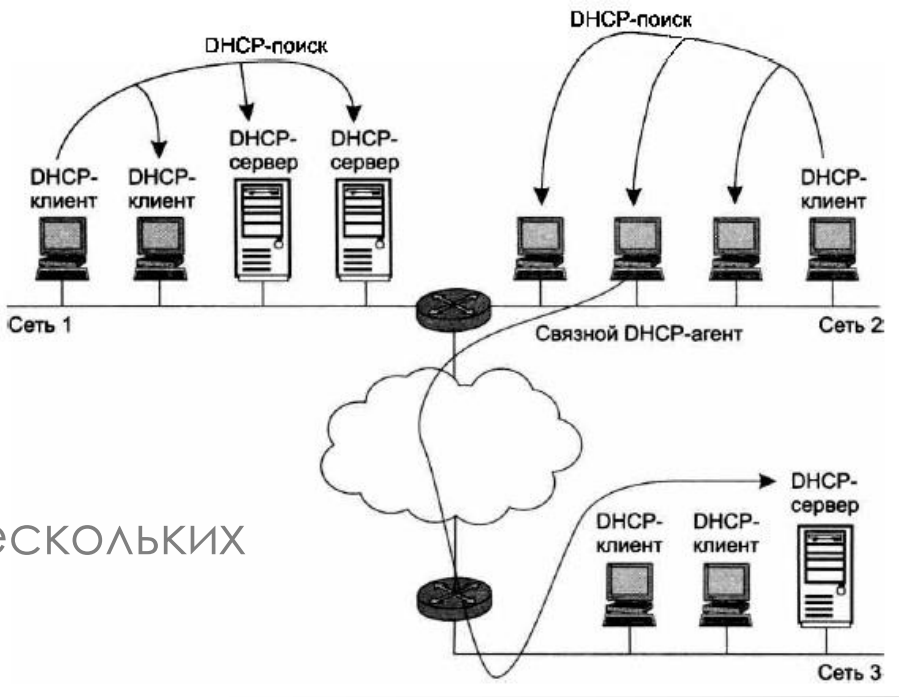
Для снижения риска выхода сети из строя из-за отказа DHCP-сервера в сети иногда ставят **резервный DHCP-сервер**.

Протокол DHCP

Если в сети нет ни одного DHCP-сервера, то его подменяет связной **DHCP-агент** – программное обеспечение, играющее роль посредника между DHCP-клиентами и DHCP-серверами.

Связной агент **переправляет** запросы клиентов из одной сети DHCP-серверу другой сети.

Один DHCP-сервер может обслуживать DHCP-клиентов нескольких разных сетей.



Протокол DHCP

Упрощенная схема обмена сообщениями между клиентскими и серверными частями DHCP:

1. Когда компьютер включают, установленный на нем DHCP-клиент посылает ограниченное широковещательное сообщение DHCP-поиска (IP-пакет с адресом назначения, состоящим из одних единиц, который должен быть доставлен всем узлам данной IP- сети) – **DHCP Discover**.

2. Находящиеся в сети DHCP-серверы получают это сообщение. Если в сети DHCP- серверы отсутствуют, то сообщение DHCP-поиска получает связной DHCP-агент. Он пересылает это сообщение в другую, возможно, значительно отстоящую от него сеть DHCP-серверу, IP-адрес которого ему заранее известен.

Протокол DHCP

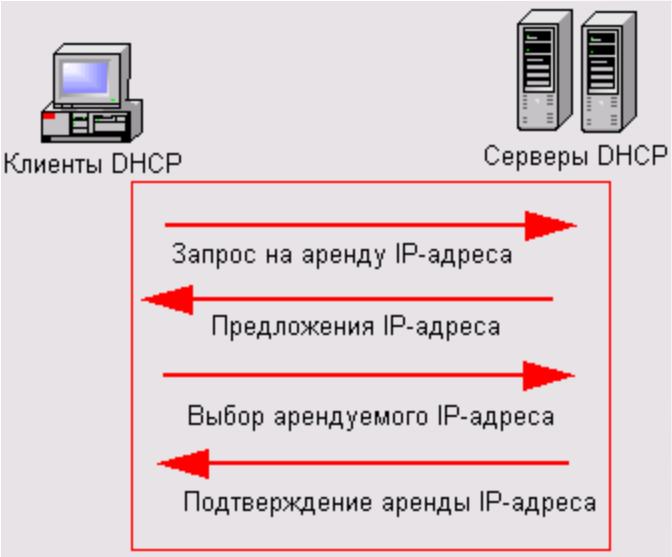
3. Все DHCP-серверы, получившие сообщение DHCP-поиска, посылают DHCP-клиенту, обратившемуся с запросом, свои DHCP-предложения (DHCP-сервер, находящийся в другой сети, посылает ответ через агента) – **DHCP Offer**. Каждое предложение содержит IP-адрес и другую конфигурационную информацию.

4. DHCP-клиент собирает конфигурационные DHCP-предложения от всех DHCP-серверов. Как правило, он выбирает первое из поступивших предложений и отправляет в сеть широковещательный DHCP-запрос – **DHCP Request**. В этом запросе содержатся идентификационная информация о DHCP-сервере, предложение которого принято, а также значения принятых конфигурационных параметров.

Протокол DHCP

5. Все DHCP-серверы получают DHCP-запрос, и только один выбранный DHCP-сервер посылает положительную DHCP-квитанцию (подтверждение IP-адреса и параметров аренды) – **DHCP ACK.** Остальные серверы аннулируют свои предложения, в частности возвращают в свои пулы предложенные адреса.

6. DHCP-клиент получает положительную DHCP-квитанцию и переходит в рабочее состояние.



Протокол DHCP

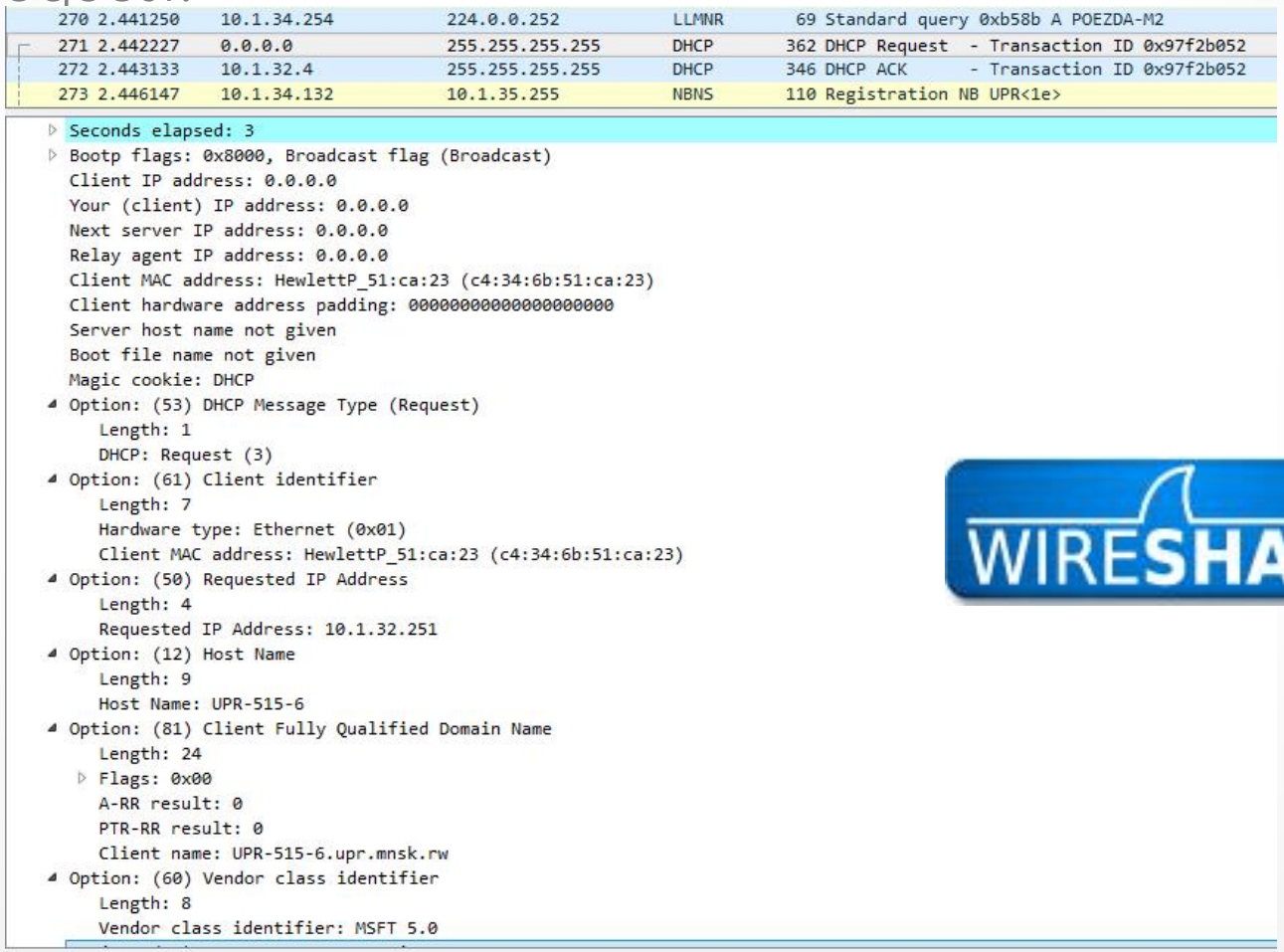
Время от времени компьютер пытается обновить параметры аренды у DHCP-сервера.

Первую попытку он делает **задолго до истечения срока аренды**, обращаясь к тому серверу, от которого он получил текущие параметры.

Если ответа нет или ответ отрицательный, он через некоторое время снова посылает запрос. Так повторяется несколько раз, и если все попытки получить параметры у того же сервера оказываются безуспешными, клиент **обращается к другому серверу**. Если и другой сервер отвечает отказом, то клиент теряет свои конфигурационные параметры и переходит в **режим автономной работы**.

Протокол DHCP

DHCP Request:

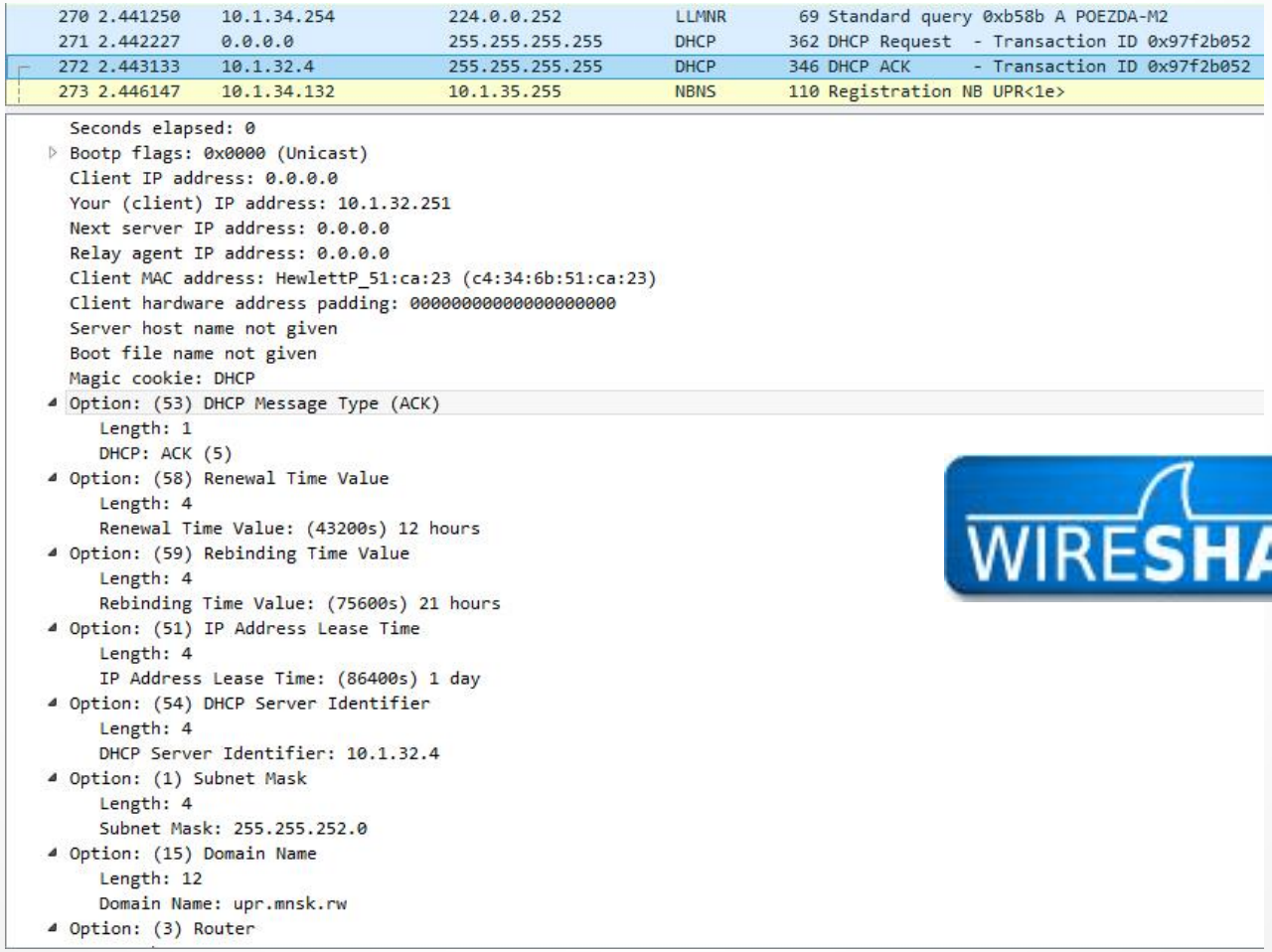


```
270 2.441250    10.1.34.254     224.0.0.252     LLMNR     69 Standard query 0xb58b A POEZDA-M2
271 2.442227    0.0.0.0         255.255.255.255 DHCP     362 DHCP Request  - Transaction ID 0x97f2b052
272 2.443133    10.1.32.4       255.255.255.255 DHCP     346 DHCP ACK      - Transaction ID 0x97f2b052
273 2.446147    10.1.34.132     10.1.35.255     NBNS     110 Registration NB UPR<1e>

▷ Seconds elapsed: 3
▷ Bootp flags: 0x8000, Broadcast flag (Broadcast)
  Client IP address: 0.0.0.0
  Your (client) IP address: 0.0.0.0
  Next server IP address: 0.0.0.0
  Relay agent IP address: 0.0.0.0
  Client MAC address: HewlettP_51:ca:23 (c4:34:6b:51:ca:23)
  Client hardware address padding: 00000000000000000000
  Server host name not given
  Boot file name not given
  Magic cookie: DHCP
◢ Option: (53) DHCP Message Type (Request)
    Length: 1
    DHCP: Request (3)
◢ Option: (61) Client identifier
    Length: 7
    Hardware type: Ethernet (0x01)
    Client MAC address: HewlettP_51:ca:23 (c4:34:6b:51:ca:23)
◢ Option: (50) Requested IP Address
    Length: 4
    Requested IP Address: 10.1.32.251
◢ Option: (12) Host Name
    Length: 9
    Host Name: UPR-515-6
◢ Option: (81) Client Fully Qualified Domain Name
    Length: 24
  ▷ Flags: 0x00
    A-RR result: 0
    PTR-RR result: 0
    Client name: UPR-515-6.upr.mnsk.rw
◢ Option: (60) Vendor class identifier
    Length: 8
    Vendor class identifier: MSFT 5.0
```

Протокол DHCP

DHCP ACK:



```
270 2.441250    10.1.34.254     224.0.0.252     LLMNR     69 Standard query 0xb58b A POEZDA-M2
271 2.442227    0.0.0.0         255.255.255.255 DHCP     362 DHCP Request  - Transaction ID 0x97f2b052
272 2.443133    10.1.32.4       255.255.255.255 DHCP     346 DHCP ACK      - Transaction ID 0x97f2b052
273 2.446147    10.1.34.132     10.1.35.255     NBNS     110 Registration NB UPR<1e>

  Seconds elapsed: 0
▷ Bootp flags: 0x0000 (Unicast)
  Client IP address: 0.0.0.0
  Your (client) IP address: 10.1.32.251
  Next server IP address: 0.0.0.0
  Relay agent IP address: 0.0.0.0
  Client MAC address: HewlettP_51:ca:23 (c4:34:6b:51:ca:23)
  Client hardware address padding: 00000000000000000000
  Server host name not given
  Boot file name not given
  Magic cookie: DHCP
◢ Option: (53) DHCP Message Type (ACK)
     Length: 1
     DHCP: ACK (5)
◢ Option: (58) Renewal Time Value
     Length: 4
     Renewal Time Value: (43200s) 12 hours
◢ Option: (59) Rebinding Time Value
     Length: 4
     Rebinding Time Value: (75600s) 21 hours
◢ Option: (51) IP Address Lease Time
     Length: 4
     IP Address Lease Time: (86400s) 1 day
◢ Option: (54) DHCP Server Identifier
     Length: 4
     DHCP Server Identifier: 10.1.32.4
◢ Option: (1) Subnet Mask
     Length: 4
     Subnet Mask: 255.255.252.0
◢ Option: (15) Domain Name
     Length: 12
     Domain Name: upr.mnsk.rw
◢ Option: (3) Router
```

Протокол DHCP

```
Адаптер беспроводной локальной сети Беспроводная сеть:

   DNS-суффикс подключения . . . . . :
   Описание. . . . . . . . . . . . . : Intel(R) Centrino(R) Wireless-N 2230 #2
   Физический адрес. . . . . . . . . : 68-17-29-CF-80-2D
   DHCP включен. . . . . . . . . . . : Да
   Автонастройка включена. . . . . . : Да
   Локальный IPv6-адрес канала . . . : fe80::c0e1:d006:e3e7:ac70%24(Основной)
   IPv4-адрес. . . . . . . . . . . . : 192.168.100.9(Основной)
   Маска подсети . . . . . . . . . . : 255.255.255.0
   Аренда получена. . . . . . . . . . : 10 октября 2017 г. 18:22:00
   Срок аренды истекает. . . . . . . . . . : 13 октября 2017 г. 21:41:38
   Основной шлюз. . . . . . . . . : fe80::1%24
                                    192.168.100.1
   DHCP-сервер. . . . . . . . . . . : 192.168.100.1
   IAID DHCPv6 . . . . . . . . . . . : 493360937
   DUID клиента DHCPv6 . . . . . . . : 00-01-00-01-19-F6-A0-33-74-D4-35-10-CD-61

   DNS-серверы. . . . . . . . . . . : 82.209.213.51
                                    82.209.213.56
   NetBios через TCP/IP. . . . . . . : Включен
```

Структура заголовка IP-пакета

Протокол IP относится к протоколам **без установления соединений**, он поддерживает обработку каждого IP-пакета как независимой единицы обмена, не связанной с другими пакетами.

Имеется **прямая связь** между **количеством полей заголовка** пакета и **функциональной сложностью протокола**, который работает с этим заголовком.

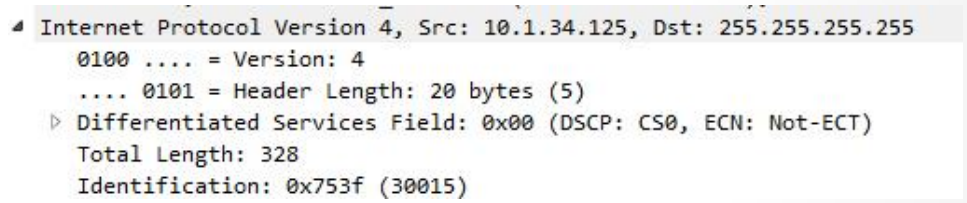
Чем проще заголовок – тем проще соответствующий протокол (большая часть действий протокола связана с обработкой той служебной информации, которая переносится в полях заголовка пакета).

Структура заголовка IP-пакета

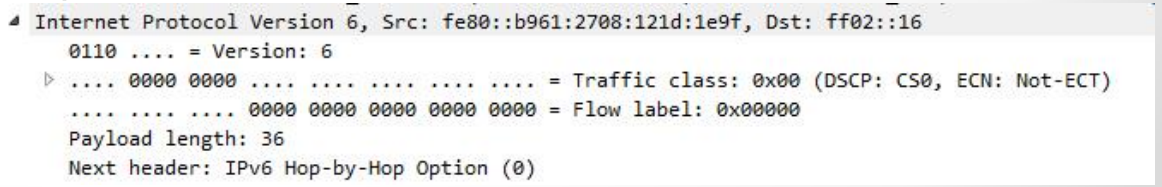
[Вернуться к заголовку IPv6 (будет понятно позже :)]

Структура заголовка IP-пакета

- Поле **номера версии** занимает 4 бита и идентифицирует версию протокола IP. Сейчас повсеместно используется **версия 4** (**IPv4**) – **0100**, хотя все чаще встречается **версия 6** (**IPv6**) – **0110**.



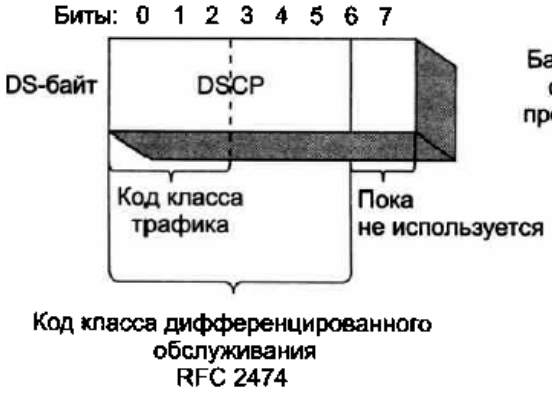
- Значение **длины заголовка** IP-пакета также занимает 4 бита и измеряется в 32-битных словах. Обычно заголовок имеет длину в **20 байт** (пять 32-битных слов), но при добавлении некоторой служебной информации это значение может быть увеличено за счет дополнительных байтов в поле параметров. **Наибольшая длина заголовка составляет 60 байт**.



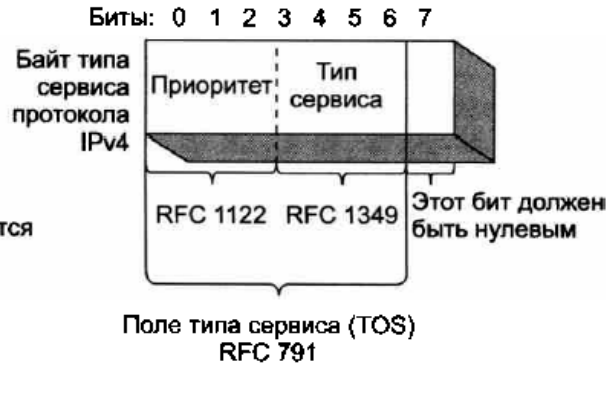
Структура заголовка IP-пакета

- Поле **типа сервиса** (**Type of Service**, **ToS**) имеет и другое, более современное название – **байт дифференцированного обслуживания,** или **DS-байт.** Этим двум названиям соответствуют два варианта интерпретации этого поля. В обоих случаях данное поле служит одной цели – хранению признаков, которые отражают требования к качеству обслуживания пакета.

  В прежнем варианте первые три бита содержат значение **приоритета** пакета: от самого низкого – 0 до самого высокого – 7.



Код класса дифференцированного обслуживания
RFC 2474



Поле типа сервиса (TOS)
RFC 791

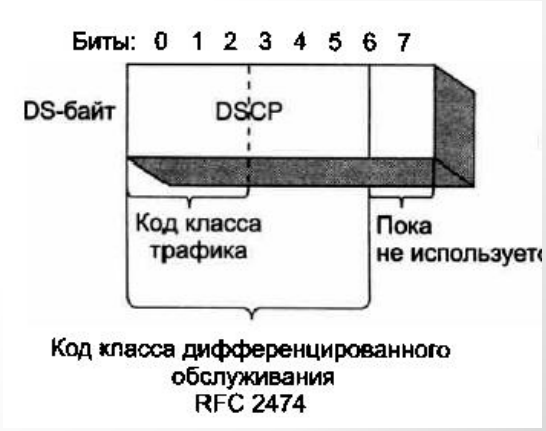
Структура заголовка IP-пакета

- Следующие три бита поля ToS определяют **критерий выбора маршрута.**
  Если бит **D** (**Delay – задержка**) установлен в 1, то маршрут должен выбираться для минимизации задержки доставки данного пакета, установленный бит **T** (**Throughput – пропускная способность**) – для максимизации пропускной способности, а бит **R** (**Reliability – надежность**) – для максимизации надежности доставки.
  Оставшиеся два бита имеют нулевое значение.

Структура заголовка IP-пакета

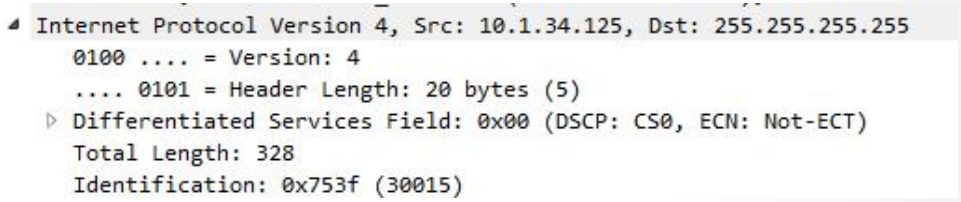
- Стандарты дифференцированного обслуживания, принятые в конце 90-х годов, дали новое название этому полю и переопределили назначение его битов.

  В настоящее время используются только старшие **шесть битов байта DS**, причем из них только старшие три требуются для определения класса трафика (восемь различных классов).
  Младший бит (из используемых шести) байта DS обычно переносит признак IN – индикатор того, что пакет «вышел» из профиля трафика и его можно отбросить или перевести в другой класс.



Структура заголовка IP-пакета

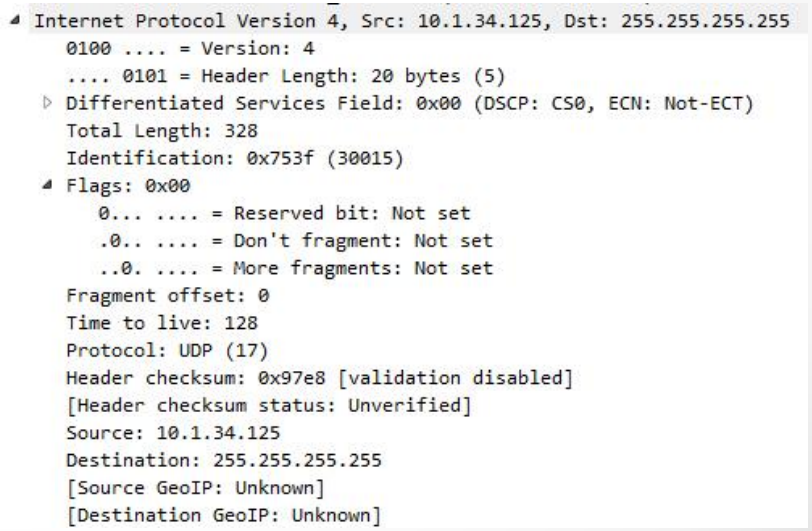
- Поле **общей длины** занимает 2 байта и характеризует общую длину пакета с учетом заголовка и поля данных. Максимальная длина пакета ограничена разрядностью поля, определяющего эту величину, и составляет **65 535 байт**, однако в большинстве компьютеров и сетей столь большие пакеты не используются.



- **Идентификатор пакета** занимает 2 байта и используется для распознавания пакетов, образовавшихся путем деления на части (фрагментации) исходного пакета. Все части (фрагменты) одного пакета должны иметь одинаковое значение этого поля.

Структура заголовка IP-пакета

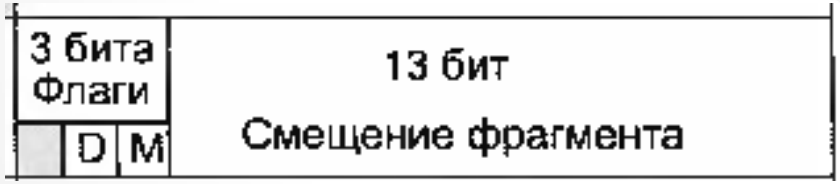
- **Флаги** занимают 3 бита и содержат признаки, связанные с фрагментацией.
  Установленный в 1 бит **DF** (**Do not Fragment – не фрагментировать**) запрещает маршрутизатору фрагментировать данный пакет, а установленный в 1 бит **MF** (**More Fragments – больше фрагментов**) говорит о том, что данный пакет является промежуточным (не последним) фрагментом.
  Оставшийся бит зарезервирован.



Вернуться к фрагментации пакета (будет понятно позже :)

Структура заголовка IP-пакета

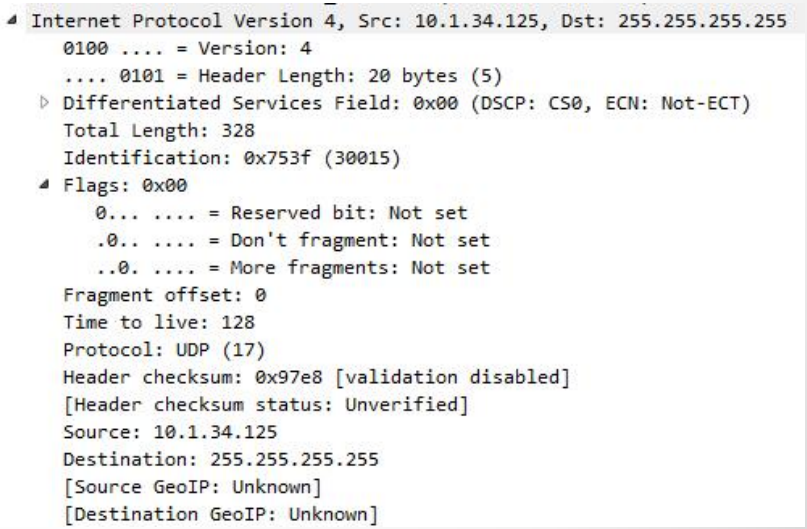
- Поле **смещения фрагмента** занимает 13 бит и задает смещение в байтах поля данных этого фрагмента относительно начала поля данных исходного (нефрагментированного) пакета.
  Используется при сборке/разборке фрагментов пакетов.
  Смещение должно быть кратно 8 байт.




```
Internet Protocol Version 4, Src: 10.1.34.125, Dst: 255.255.255.255
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▷ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 328
    Identification: 0x753f (30015)
  ◢ Flags: 0x00
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 128
    Protocol: UDP (17)
    Header checksum: 0x97e8 [validation disabled]
    [Header checksum status: Unverified]
    Source: 10.1.34.125
    Destination: 255.255.255.255
    [Source GeoIP: Unknown]
    [Destination GeoIP: Unknown]
```

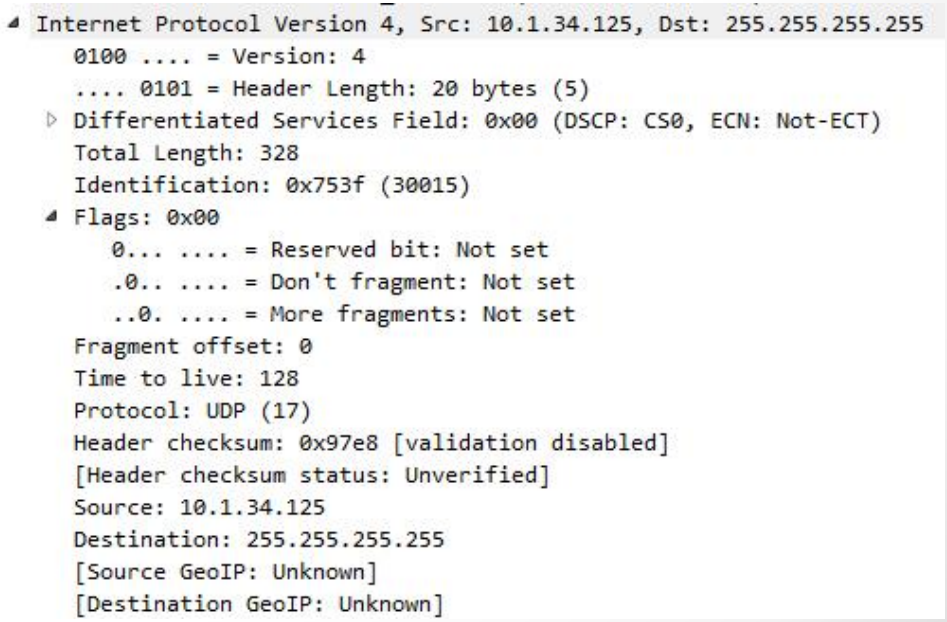
Структура заголовка IP-пакета

- Поле **времени жизни** (**Time To Live**, **TTL**) занимает один байт и используется для задания **предельного срока**, в течение которого пакет может перемещаться по сети.
  Время жизни пакета измеряется в секундах и задается источником.
  По истечении каждой секунды пребывания на каждом из маршрутизаторов, через которые проходит пакет во время своего «путешествия» по сети, из его текущего времени жизни вычитается единица; единица вычитается и в том случае, если время пребывания было **меньше секунды**.



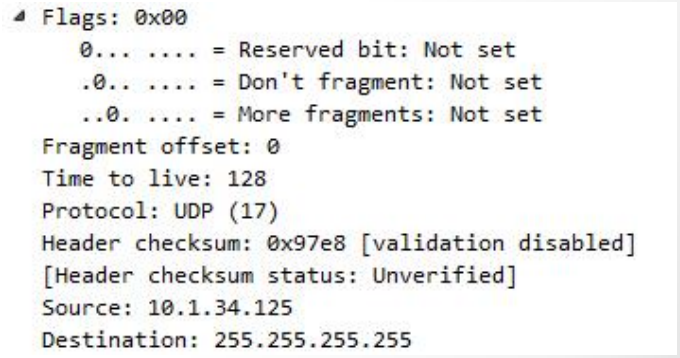
Структура заголовка IP-пакета

- Поле **протокола верхнего уровня** занимает один байт и содержит идентификатор, указывающий, какому протоколу верхнего уровня принадлежит информация, размещенная в поле данных пакета:
  1 – ICMP;
  2 – IGMP;
  4 – IP;
  6 – TCP;
  17 –UDP.



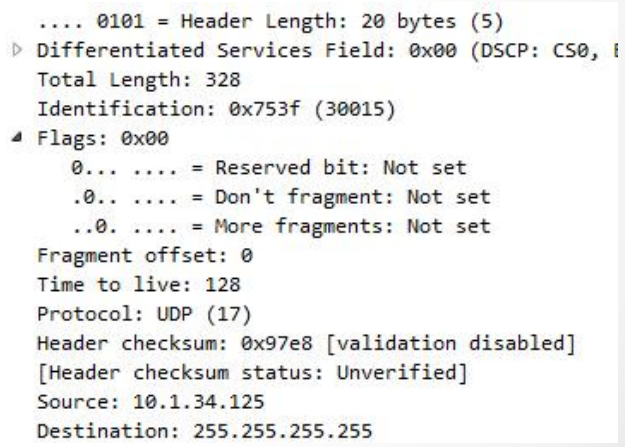
Структура заголовка IP-пакета

- **Контрольная сумма заголовка** занимает 2 байта (16 бит) и рассчитывается только по заголовку.
  Поскольку некоторые поля заголовка **меняют свое значение** в процессе передачи пакета по сети (например, поле времени жизни), контрольная сумма проверяется и повторно рассчитывается на **каждом** маршрутизаторе и конечном узле как дополнение к сумме всех 16-битных слов заголовка.
  При вычислении контрольной суммы значение самого поля контрольной суммы устанавливается в **нуль**.
  Если **контрольная сумма неверна**, то пакет **отбрасывается**, как только обнаруживается ошибка.



Структура заголовка IP-пакета

- Поля **IP-адресов источника** и **приемника** имеют одинаковую длину – 32 бита.
- Поле **параметров** состоит из нескольких подполей одного из восьми предопределенных типов и является не обязательным (используется обычно только при отладке сети).
  В этих подполях можно указывать точный маршрут (**маршрутизация от источника)**, регистрировать проходимые пакетом маршрутизаторы или помещать данные системы безопасности и временные отметки.
  Заголовок должен быть **выравнен** нулевыми байтами по **32-битной границе**.



Маршрутизаторы

Маршрутизатор – специализированный сетевой компьютер, имеющий два или более сетевых интерфейсов и пересылающий пакеты данных между различными сегментами сети.

Маршрутизатор может связывать разнородные сети различных архитектур.

Для принятия решений о пересылке пакетов протоколом маршрутизации используется **информация о топологии сети**, **критерии выбора маршрута** и определённые **правила**, **заданные администратором**.

Маршрутизаторы

В качестве критерия может выступать:

- **задержка** прохождения маршрута отдельным пакетом;
- **средняя пропускная способность** маршрута для последовательности пакетов;
- **количество пройденных** на маршруте **промежуточных маршрутизаторов** (ретрансляционных участков, или хопов).

Полученная в результате анализа информация о маршрутах дальнейшего следования пакетов помещается в **таблицу маршрутизации**.

Таблицы маршрутизации

Первый столбец таблицы содержит **адреса назначения пакетов.**

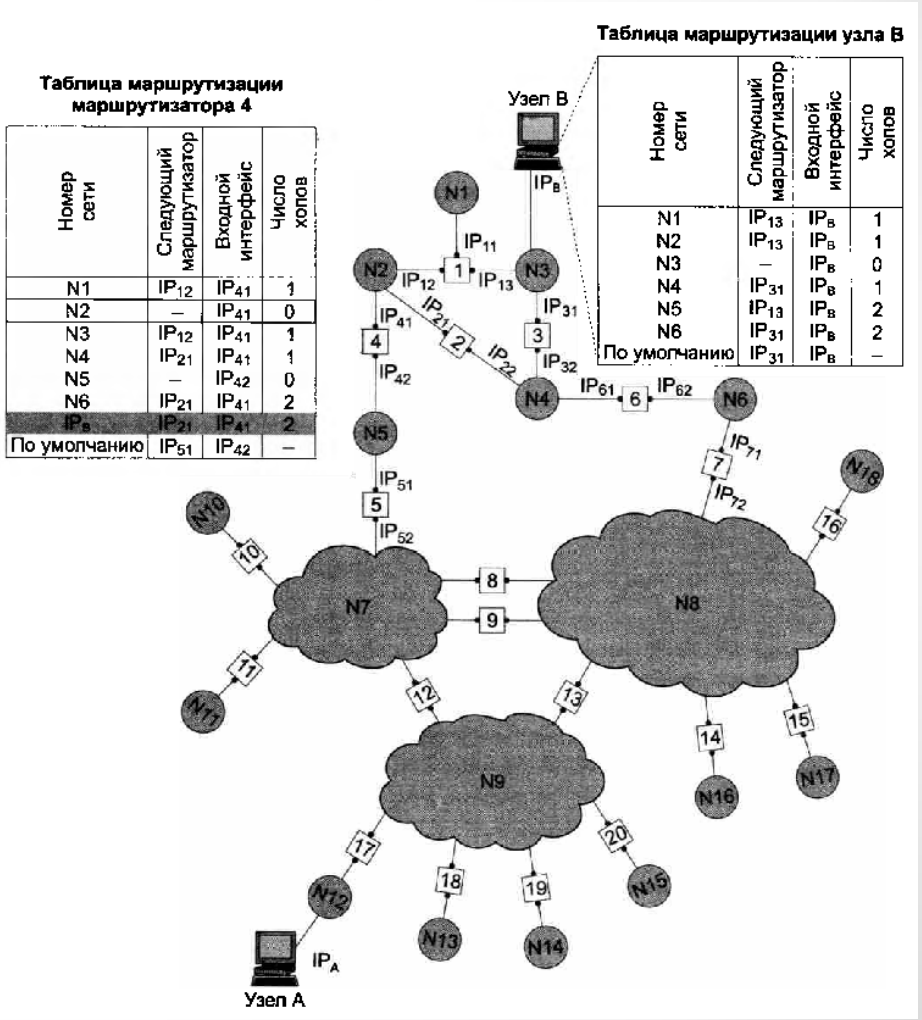
Во втором столбце указывается **сетевой адрес интерфейса следующего маршрутизатора**, на который надо направить пакет, чтобы тот передвигался по направлению к заданному адресу по рациональному маршруту.

Таблица маршрутизации маршрутизатора 4

Номер сети	Следующий маршрутизатор	Входной интерфейс	Число хопов
N1	IP_{12}	IP_{41}	1
N2	–	IP_{41}	0
N3	IP_{12}	IP_{41}	1
N4	IP_{21}	IP_{41}	1
N5	–	IP_{42}	0
N6	IP_{21}	IP_{41}	2
IP_B	IP_{21}	IP_{41}	2
По умолчанию	IP_{51}	IP_{42}	–

Таблица маршрутизации узла B

Номер сети	Следующий маршрутизатор	Входной интерфейс	Число хопов
N1	IP_{13}	IP_B	1
N2	IP_{13}	IP_B	1
N3	–	IP_B	0
N4	IP_{31}	IP_B	1
N5	IP_{13}	IP_B	2
N6	IP_{31}	IP_B	2
По умолчанию	IP_{31}	IP_B	–



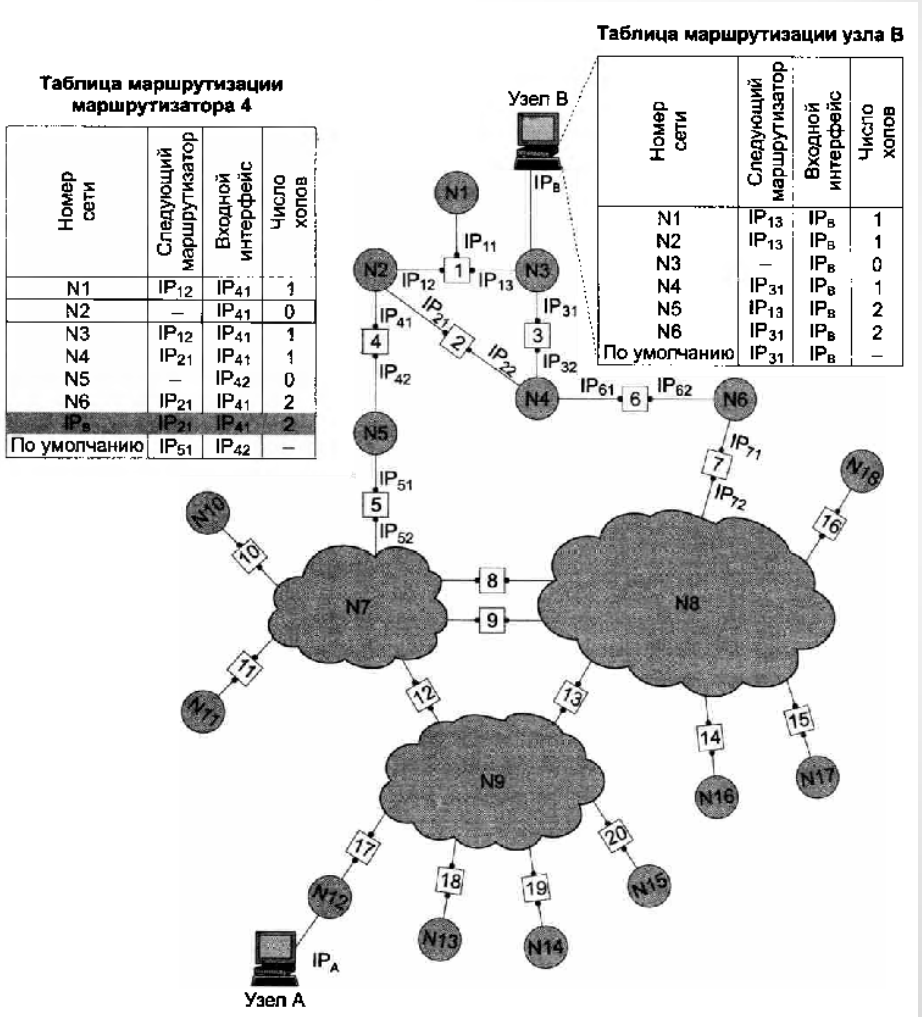
Таблицы маршрутизации

В третьем столбце указан один из нескольких **собственных портов** в который должен быть помещен пакет для передачи.

В последнем столбце указано **расстояние до адресата** – это необходимо при выборе одной из нескольких записей в таблице.

Таблица маршрутизации маршрутизатора 4

Номер сети	Следующий маршрутизатор	Входной интерфейс	Число хопов
N1	IP_{12}	IP_{41}	1
N2	–	IP_{41}	0
N3	IP_{12}	IP_{41}	1
N4	IP_{21}	IP_{41}	1
N5	–	IP_{42}	0
N6	IP_{21}	IP_{41}	2
IP_B	IP_{21}	IP_{41}	2
По умолчанию	IP_{51}	IP_{42}	–



Таблицы маршрутизации

Определение направления продвижения пакета по таблице маршрутизации:

1. На один из интерфейсов маршрутизатора поступает пакет. Протокол IP извлекает из пакета IP-адрес его назначения.

2. Выполняется **первая фаза** просмотра таблицы – **поиск конкретного маршрута к узлу**. IP-адрес (целиком) последовательно строка за строкой сравнивается с содержимым поля адреса назначения таблицы маршрутизации. Если произошло совпадение, то из соответствующей строки извлекаются адрес следующего маршрутизатора и идентификатор выходного интерфейса. На этом просмотр таблицы заканчивается.

Таблицы маршрутизации

Определение направления продвижения пакета по таблице маршрутизации (продолжение):

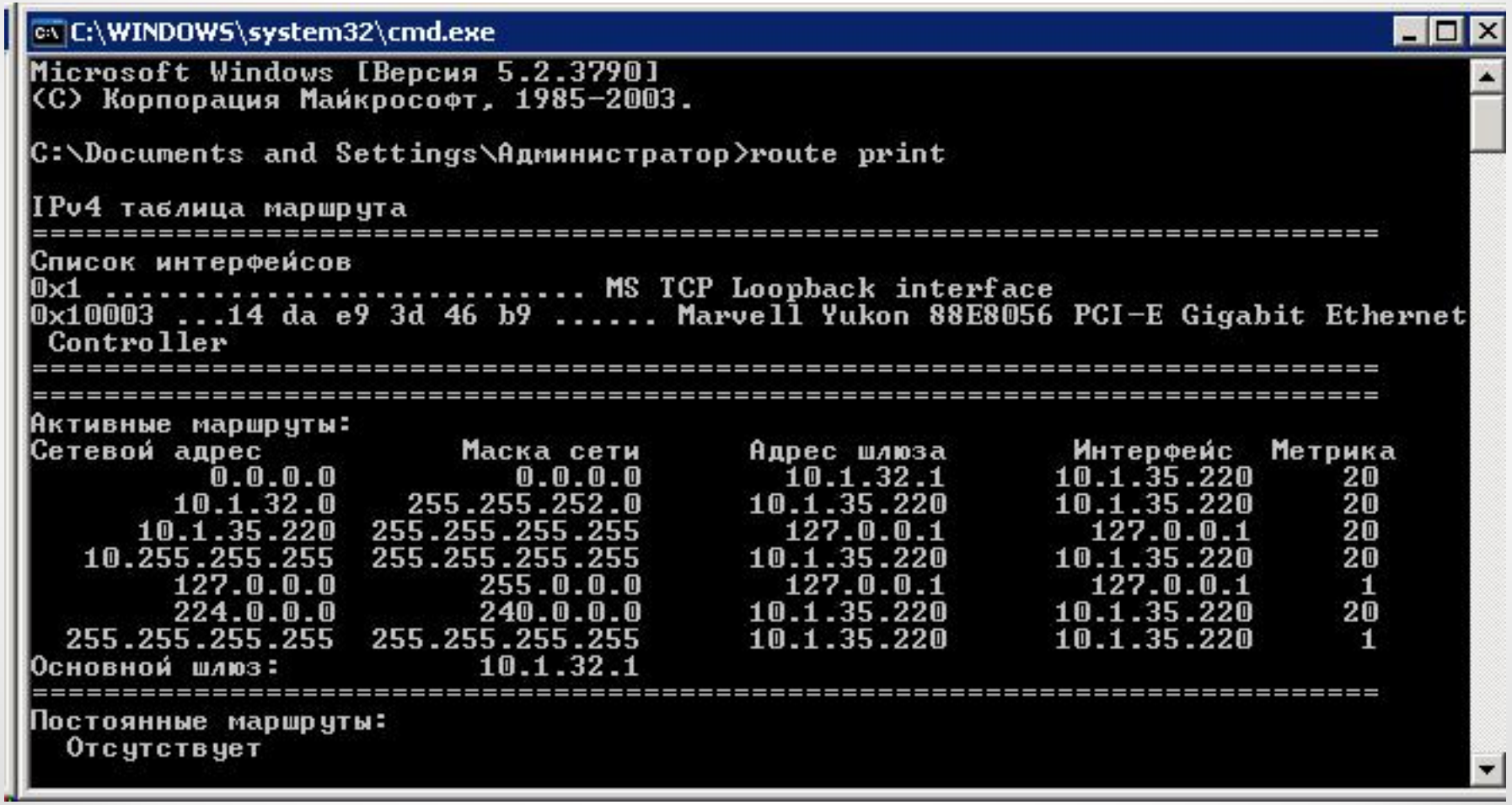
3. Если в таблице нет строки с адресом назначения, то протокол IP переходит ко **второй фазе** просмотра – **поиску маршрута к сети назначения**. Из IP-адреса выделяется номер сети, и таблица снова просматривается на предмет совпадения номера сети в какой-либо строке с номером сети из пакета. При совпадении из соответствующей строки таблицы извлекаются адрес следующего маршрутизатора и идентификатор выходного интерфейса. Просмотр таблицы на этом завершается.

Таблицы маршрутизации

Определение направления продвижения пакета по таблице маршрутизации (продолжение):

4. Если совпадения не произошло ни в первой, ни во второй фазе просмотра, то средствами протокола IP либо выбирается **маршрут по умолчанию**, либо, если маршрут по умолчанию отсутствует, **пакет отбрасывается**. Просмотр таблицы на этом заканчивается.

Таблицы маршрутизации



Таблицы маршрутизации

Практически для всех маршрутизаторов существуют **три основных источника записей в таблице**:

- **программное обеспечение стека TCP/IP**, которое при инициализации маршрутизатора автоматически заносит в таблицу несколько записей, в результате чего создается так называемая **минимальная таблица маршрутизации** (для MS Windows:127.0.0.0; 0.0.0.0; 198.21.17.255; 213.34.12.255; 224.0.0.0; 255.255.255.255; для аппаратного: 198.21.17.0; 213.34.12.0);
- **администратор**, непосредственно формирующий **статические записи** с помощью некоторой системной утилиты (программа route);
- **протоколы маршрутизации** (RIP, OSPF, BGP, EIGRP), формирующие **динамические записи**, имеющие ограниченный срок жизни.

Таблицы маршрутизации

Для выбора оптимального маршрута при наличии **двух и более** различных маршрутов до одной цели по различным протоколам маршрутизации используется **административное расстояние** – это степень надежности источника маршрутной информации (протоколы динамической маршрутизации, статические маршруты и непосредственно подключенные сети).

Каждому протоколу маршрутизации назначается **приоритет надежности** (**достоверности**), от максимального до минимального, указанный с помощью значения **административного расстояния**.

Таблицы маршрутизации

Значения **административных расстояний** по умолчанию для некоторых протоколов, поддерживаемых Cisco:

Источник маршрута	Административное расстояние
Подключенный интерфейс (Connected)	0
Статический маршрут (Static)	1
Объединенный маршрут по протоколу EIGRP (EIGRP summary)	5
Внешний протокол BGP (eBGP)	20
Внутренний протокол EIGRP (iEIGRP)	90
Протокол IGRP	100
Протокол OSPF	110
Протокол IS-IS	115
Протокол RIP	120
Внешний протокол EIGRP (eEIGRP)	170
Внутренний протокол BGP (iBGP)	200

Протокол RIP

Протокол **RIP** (**Routing Information Protocol – протокол маршрутной информации**) является внутренним протоколом маршрутизации дистанционно-векторного типа.

Этот протокол прост в реализации и чаще всего используется в небольших сетях.

Для IP имеются две версии RIP – **RIPv1** (не поддерживает масок) и **RIPv2** (передает информацию о масках сетей).

Протокол RIP

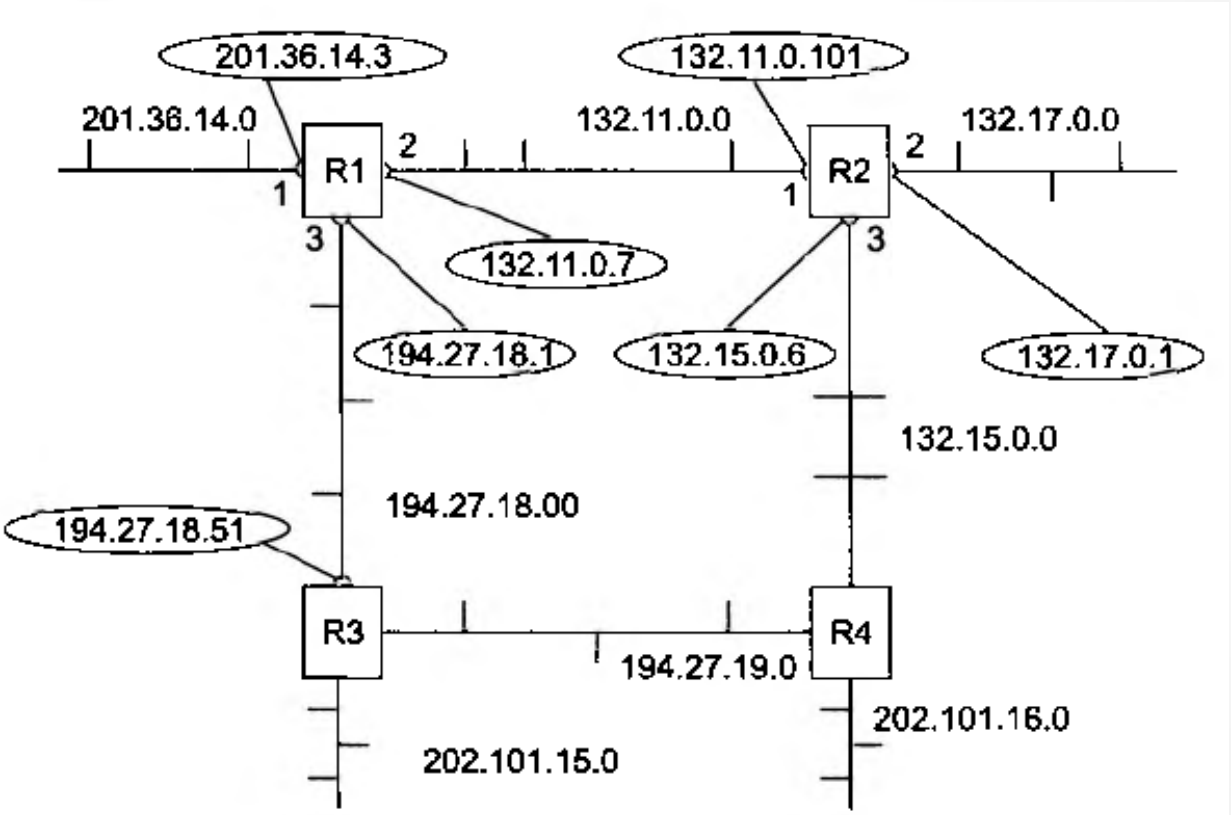
Для измерения расстояния до сети стандарты протокола RIP допускают различные виды **метрик**:

- **хопы**;
- **значения пропускной способности**;
- **вносимые задержки**;
- **надежность сетей** (соответствующие признакам D, T и R в поле качества сервиса IP-пакета);
- **любые комбинации этих метрик**.

Метрика должна обладать свойством **аддитивности** – метрика составного пути должна быть равна сумме метрик составляющих этого пути.

Протокол RIP

Сеть включает **восемь** IP-сетей, связанных **четырьмя** маршрутизаторами с идентификаторами: **R1**, **R2**, **R3** и **R4**.



Протокол RIP

Построение таблицы маршрутизации:

1. **Создание минимальной таблицы**. В исходном состоянии на каждом маршрутизаторе программным обеспечением стека TCP/ IP автоматически создается минимальная таблица маршрутизации, в которой учитываются только непосредственно подсоединенные сети.

Минимальная таблица маршрутизации R1:

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	201.36.14.3	1	1
132.11.0.0	132.11.0.7	2	1
194.27.18.0	194.27.18.1	3	1

Протокол RIP

2. Рассылка минимальной таблицы соседям. После инициализации каждый маршрутизатор начинает посылать своим соседям сообщения протокола RIP, в которых содержится его минимальная таблица.

RIP-сообщения передаются в дейтаграммах протокола UDP и включают два параметра для каждой сети:

- ее **IP-адрес**;
- **расстояние до нее** от передающего сообщение маршрутизатора.

Маршрутизатор **R1** передает маршрутизаторам **R2** и **R3** следующие сообщения:

сеть 201.36.14.0, **расстояние 1**; **сеть 132.11.0.0**, **расстояние 1**; **сеть 194.27.18.0**, **расстояние 1**.

Протокол RIP

3. Получение RIP-сообщений от соседей и обработка полученной информации. После получения аналогичных сообщений от маршрутизаторов R2 и R3 маршрутизатор R1 наращивает каждое полученное поле метрики на единицу и запоминает, через какой порт и от какого маршрутизатора получена новая информация (адрес этого маршрутизатора станет адресом следующего маршрутизатора, если эта запись будет внесена в таблицу маршрутизации).

Затем маршрутизатор начинает сравнивать новую информацию с той, которая хранится в его таблице маршрутизации.

Протокол RIP

Таблица маршрутизации R1:

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	201.36.14.3	1	1
132.11.0.0	132.11.0.7	2	1
194.27.18.0	194.27.18.1	3	1
132.17.0.0	132.11.0.101	2	2
132.15.0.0	132.11.0.101	2	2
194.27.19.0	194.27.18.51	3	2
202.101.15.0	194.27.18.51	3	2
~~132.11.0.0~~	~~132.11.0.101~~	~~2~~	~~2~~
~~194.27.18.0~~	~~194.27.18.51~~	~~3~~	~~2~~

Протокол RIP

4. Рассылка новой таблицы соседям. Каждый маршрутизатор отсылает новое RIP- сообщение всем своим соседям.

В этом сообщении он помещает данные обо всех известных ему сетях: как непосредственно подключенных, так и удаленных, о которых маршрутизатор узнал из RIP-сообщений.

5. Получение RIP-сообщений от соседей и обработка полученной информации. Этап 5 повторяет этап 3 – маршрутизаторы принимают RIP-сообщения, обрабатывают содержащуюся в них информацию и на ее основании корректируют свои таблицы маршрутизации.

Протокол RIP

Таблица маршрутизации R1:

Номер сети	Адрес следующего маршрутизатора	Порт	Расстояние
201.36.14.0	201.36.14.3	1	1
132.11.0.0	132.11.0.7	2	1
194.27.18.0	194.27.18.1	3	1
132.17.0.0	132.11.0.101	2	2
132.15.0.0	132.11.0.101	2	2
~~132.15.0.0~~	~~194.27.18.51~~	~~3~~	~~3~~
194.27.19.0	194.27.18.51	3	2
~~194.27.19.0~~	~~132.11.0.101~~	~~2~~	~~3~~
~~202.101.15.0~~	~~194.27.18.51~~	~~3~~	~~2~~
202.101.16.0	132.11.0.101	2	3
~~202.101.16.0~~	~~194.27.18.51~~	~~3~~	~~3~~

Протокол RIP

Если маршрутизаторы периодически повторяют этапы рассылки и обработки RIP- сообщений (каждые **30 секунд**), то за конечное время в сети установится **корректный режим маршрутизации** (состояние таблиц маршрутизации, когда все сети достижимы из любой сети с помощью некоторого рационального маршрута).

В сетях **постоянно** происходят изменения – меняется работоспособность маршрутизаторов и линий связи, маршрутизаторы и линии связи могут добавляться в существующую сеть или же выводиться из ее состава.

Протокол RIP

К новым маршрутам маршрутизаторы RIP приспосабливаются **просто** – они передают новую информацию в очередном сообщении своим соседям, и постепенно эта информация становится известна всем маршрутизаторам сети.

К изменениям, связанным с потерей какого-либо маршрута, маршрутизаторы RIP адаптируются **сложнее**.

Для уведомления о том, что некоторый маршрут недействителен, используются два механизма:

- **истечение времени жизни маршрута**;
- **указание специального** (**бесконечного**) **расстояния до сети**, **ставшей недоступной**.

Протокол RIP

Механизм **истечения времени жизни маршрута** основан на том, что каждая запись таблицы маршрутизации (как и записи таблицы продвижения моста/коммутатора), полученная по протоколу RIP, имеет **время жизни** (**TTL**).

При поступлении очередного RIP-сообщения, которое подтверждает справедливость данной записи, таймер времени жизни устанавливается в исходное состояние (**шестикратное** значение периода рассылки – **180 секунд**), а затем из него каждую секунду вычитается единица.

Если за время тайм-аута не придет новое сообщение об этом маршруте, он помечается как **недействительный**.

Протокол RIP

Механизм **указание специального** (**бесконечного**) **расстояния до сети**, **ставшей недоступной**, используется тогда, когда маршрутизатор, непосредственно примыкающий к недоступной сети, может послать сообщение о маршруте к данной сети.

Для этого маршрутизатор принудительно выставляет **бесконечное число хопов** до данной сети в своей рассылке (в протоколе RIP бесконечным условно считается расстояние в
16 хопов).

Протокол RIP

Борьба с ложными маршрутами.

Протокол RIP не в состоянии полностью исключить в сети переходные состояния, когда некоторые маршрутизаторы пользуются устаревшей информацией о несуществующих маршрутах.

Проблема с петлей, образующейся между соседними маршрутизаторами, надежно решается с помощью метода **расщепления горизонта**.

Этот метод заключается в том, что маршрутная информация о некоторой сети, хранящаяся в таблице маршрутизации, **никогда не передается** тому маршрутизатору, от которого она получена.

Протокол RIP

Расщепление горизонта не помогает в тех случаях, когда петли образуются **не двумя**, а **большим числом маршрутизаторов**.

Для **предотвращения зацикливания пакетов** по составным петлям при отказах связей применяются:

- **триггерные обновления**;
- **замораживание изменений**.

Протокол RIP

Прием **триггерных обновлений** состоит в том, что маршрутизатор, получив данные об изменении метрики до какой-либо сети, не ждет истечения периода передачи таблицы маршрутизации, а передает данные об изменившемся маршруте **немедленно**, что помогает предотвратить передачу устаревших сведений об отказавшем маршруте, но перегружает сеть служебными сообщениями, поэтому триггерные объявления также делаются с некоторой задержкой.

Протокол RIP

Прием **замораживания изменений** связан с введением тайм-аута на принятие новых данных о сети, которая только что стала недоступной.

Этот тайм-аут предотвращает принятие устаревших сведений о некоем маршруте от тех маршрутизаторов, которые находятся на некотором расстоянии от отказавшей связи и передают устаревшие сведения о ее работоспособности.

Предполагается, что в течение тайм-аута «замораживания изменений» эти маршрутизаторы **вычеркнут** данный маршрут из своих таблиц, так как не получат о нем новых записей и не будут распространять устаревшие сведения по сети.

Протокол OSPF

Протокол OSPF (**Open Shortest Path First – выбор кратчайшего пути первым**) является последним протоколом, основанном на алгоритме состояния связей, и обладает многими особенностями, ориентированными на применение в больших гетерогенных сетях.

Протокол OSPF

Построение таблицы маршрутизации:

1. Построение и поддержание базы данных о состоянии связей сети. Связи сети могут быть представлены в виде **графа**, в котором вершинами графа являются маршрутизаторы и подсети, а ребрами – связи между ними.

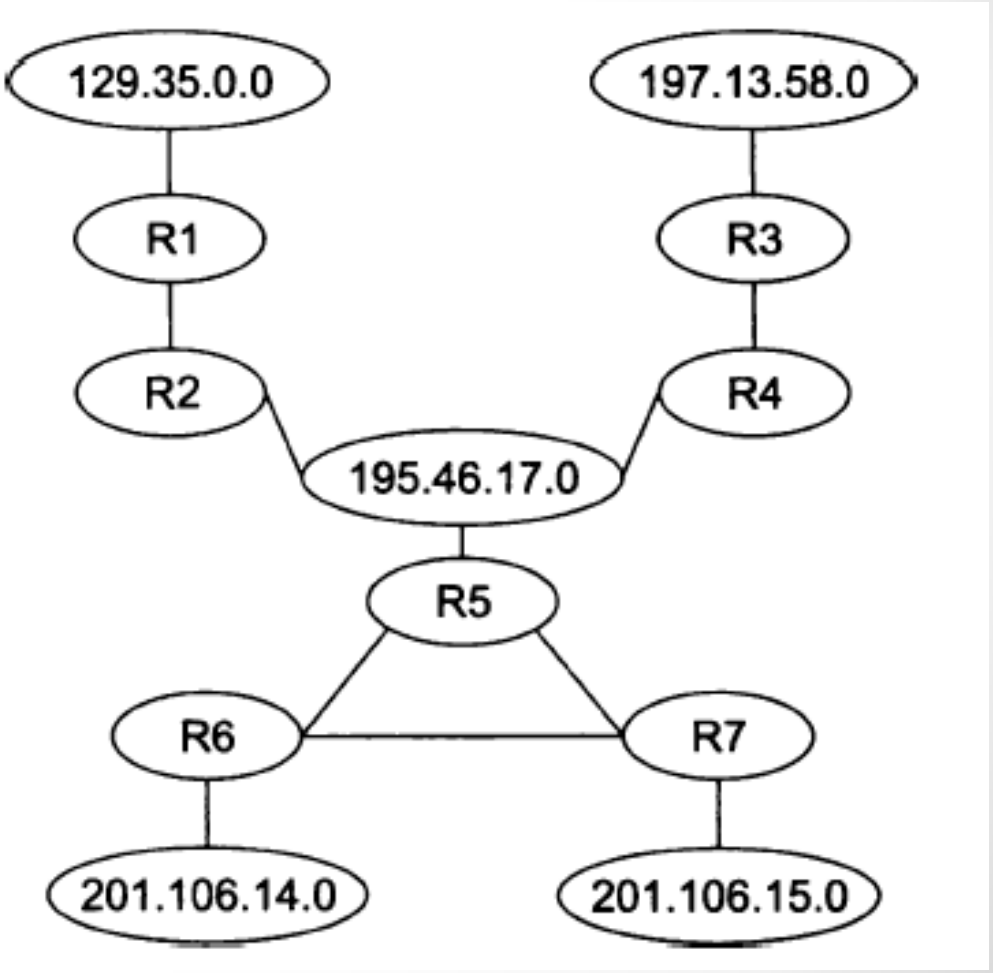
Каждый маршрутизатор обменивается со своими соседями той информацией о графе сети, которой он располагают к данному моменту (информация только о **топологии** сети).

Сообщения, с помощью которых распространяется топологическая информация, называются **объявлениями о состоянии связей** (**Link State Advertisement**, **LSA**) сети.

Протокол OSPF

При транзитной передаче объявлений LSA маршрутизаторы **не модифицируют** информацию, а передак ее в неизменном виде.

В результате все маршрутизаторы сети сохраняют в своей памяти **идентичные сведения** о текущей конфигурации графа связей сети.



Протокол OSPF

Для контроля состояния связей и соседних маршрутизаторов маршрутизаторы OSPF передают друг другу особые **сообщения HELLO** каждые **10 секунд**.

Небольшой объем этих сообщений делает возможным частое тестирование состояния соседей и связей с ними.

В том случае, когда **сообщения HELLO перестают поступать** от какого-либо непосредственного соседа, маршрутизатор делает вывод о том, что **состояние связи изменилось** с работоспособного на неработоспособное, **вносит** соответствующие **коррективы** в свою топологическую базу данных и **рассылает информацию** об этом своим соседям.

Протокол OSPF

Построение таблицы маршрутизации (продолжение):

2. Нахождение оптимальных маршрутов и генерация таблицы маршрутизации. Для этого используется **итеративный алгоритм Дейкстры**.

Каждый маршрутизатор сети, действуя в соответствии с этим алгоритмом, ищет **оптимальные маршруты** от своих интерфейсов до всех известных ему подсетей.

В каждом найденном таким образом маршруте запоминается только **один шаг – до следующего маршрутизатора**.

Данные об этом шаге и попадают в **таблицу маршрутизации**.

Протокол OSPF

Если состояние связей в сети изменилось и произошла корректировка графа сети, каждый маршрутизатор **заново** ищет оптимальные маршруты и корректирует свою таблицу маршрутизации.

Аналогичный процесс происходит и в том случае, когда в сети появляется **новая связь** или **новый сосед**, объявляющий о себе с помощью своих сообщений HELLO.

Когда состояние сети не меняется, то **объявления о связях не генерируются**, что экономит пропускную способность сети и вычислительные ресурсы маршрутизаторов.

Однако каждые **30 минут** маршрутизаторы OSPF обмениваются данными о топологии сети для надежности.

Протокол OSPF

При поиске оптимальных маршрутов протокол OSPF по умолчанию использует метрику, **учитывающую пропускную способность каналов связи**, однако допускается применение метрик, учитывающих задержки и надежность передачи пакетов каналами связи.

Для каждой из метрик протокол OSPF строит **отдельную** таблицу маршрутизации.

Выбор нужной таблицы происходит в зависимости от значений битов **TOS** в заголовке пришедшего IP-пакета.

Протокол OSPF

Вычислительная сложность протокола OSPF быстро растет с увеличением размера сети.

Для преодоления этого недостатка в протоколе OSPF вводится понятие **области сети.**

Маршрутизаторы, принадлежащие некоторой области, строят граф связей только для этой области.

Между областями информация о связях не передается, а пограничные для областей маршрутизаторы обмениваются только информацией об адресах сетей, имеющихся в каждой из областей, и **расстоянием от пограничного маршрутизатора до каждой сети**.

Протокол EIGRP

Протокол **EIGRP** (**Enhanced Interior Gateway Routing Protocol**) – протокол маршрутизации, разработанный фирмой **Cisco** на основе протокола **IGRP** той же фирмы.

Более ранний и практически не используемый ныне протокол **IGRP** был создан как альтернатива протоколу **RIP**.

EIGRP появился после **OSPF** и благодаря специальному алгоритму распространения информации об изменениях в топологии сети защищен от особых ситуаций с зацикливанием маршрутов.

EIGRP более прост в реализации и менее требователен к вычислительным ресурсам маршрутизатора чем **OSPF**.

Протокол EIGRP

EIGRP имеет более сложный алгоритм вычисления метрики **DUAL** (**Diffusing Update ALgorithm**), который может использовать 5 различных компонентов для расчета:

- **пропускная способность** (**Bandwidth**) – минимальная пропускная способность для маршрута (а не сумма в отличие от OSPF);
- **задержка** (**Delay**) – суммарная задержка на всём пути маршрута;
- **надежность** (**Reliability**) – наихудший показатель надёжности на всём пути маршрута;
- **загруженность** (**Loading**) – наихудший показатель загруженности интерфейса на всём пути маршрута (количество трафика проходящего через интерфейс при настроенном на нём параметре bandwidth);
- **минимальный размер MTU на всём пути маршрута**.

Протокол EIGRP

Протокол **EIGRP** гибридный – объединяет свойства дистанционно-векторных протоколов и протоколов по состоянию канала.

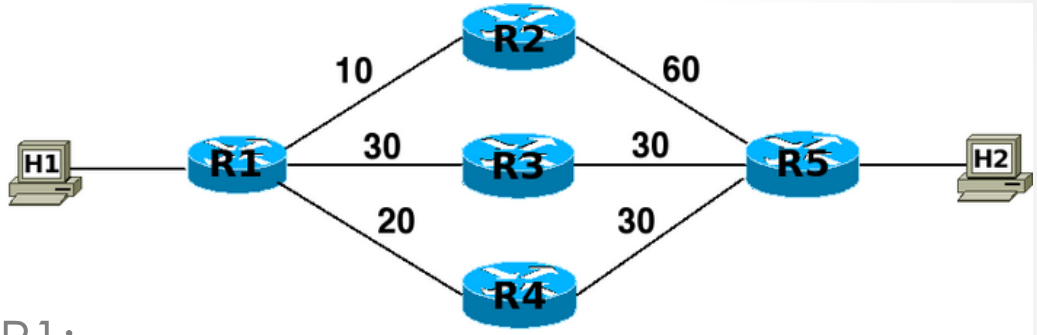


Таблица маршрутизации R1:

Соседний маршрутизатор	Роль соседнего маршрутизатора	Статус маршрута	FD	AD	Feasible condition
R2	Не выбран	Не выбран	R1R2+R2R5 = 70	R2R5 = 60	Не выполняется
R3	Feasible successor	Резервный маршрут	R1R3+R3R5 = 60	R3R5 = 30	Выполняется AD < FD (R3R5 < R1R4R5)
R4	Successor	*Лучший маршрут*	R1R4+R4R5 = 50	R4R5 = 30	*Лучший маршрут*

Протокол EIGRP

Advertised distance (AD) – стоимость расстояния между соседним маршрутизатором, который анонсирует маршрут, и сетью назначения.

Feasible distance (FD) – стоимость расстояния от локального маршрутизатора до сети назначения = AD, которое анонсирует соседний маршрутизатор + стоимость расстояния между локальным маршрутизатором и соседним маршрутизатором.

Successor – соседний маршрутизатор с путем без петель и с наименьшей стоимостью пути к сети назначения.

Feasible successor – резервный маршрутизатор с путем без петель (**Feasible condition** – AD feasible successor должно быть меньше, чем FD текущего маршрута successor).

Протокол EIGRP

EIGRP использует 5 типов сообщений:

- **Hello** – маршрутизаторы используют широковещательные hello-пакеты без подтверждения для обнаружения соседей;
- **Update** – сообщения, в которых содержится информация об изменении маршрутов, отправляются только маршрутизаторам, которых касается обновление (получение update-пакета подтверждается отправкой **ACK**);
- **Query** – когда маршрутизатор выполняет подсчет маршрута и у него нет feasible successor, он отправляет query-пакет своим соседям для того чтобы определить нет ли feasible successor для этого пункта назначения у них (получение query-пакета подтверждается отправкой **ACK**);

Протокол EIGRP

EIGRP использует 5 типов сообщений (продолжение):

- **Reply** – маршрутизатор отправляет reply-пакет в ответ на query-пакет (получение reply-пакета подтверждается отправкой **ACK**);
- **ACK** – пакет, который подтверждает получение пакетов **update**, **query**, **reply**.

Протокол BGP

Пограничный (внешний) шлюзовой протокол (**Border Gateway Protocol**, **BGP**) в версии 4 является сегодня основным протоколом обмена маршрутной информацией между автономными системами Интернета.

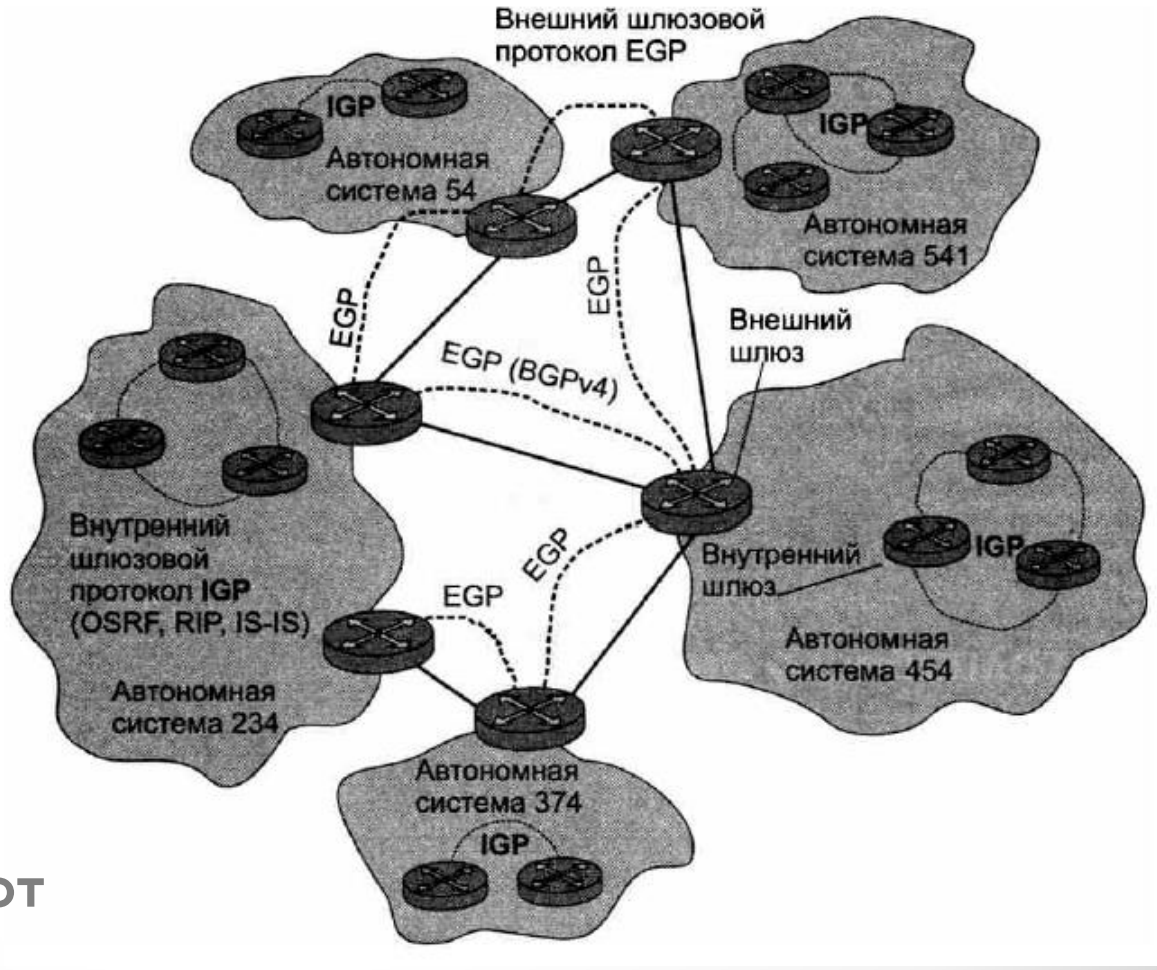
Автономная система (**Autonomous System**, **AS**) – это совокупность сетей под единым административным управлением, обеспечивающим общую для всех входящих в автономную систему маршрутизаторов политику маршрутизации.

Внешние шлюзы предназначены для соединения между собой автономных систем, каждая из которых состоит из взаимосвязанных сетей.

Протокол BGP

Автономные системы Интернета.

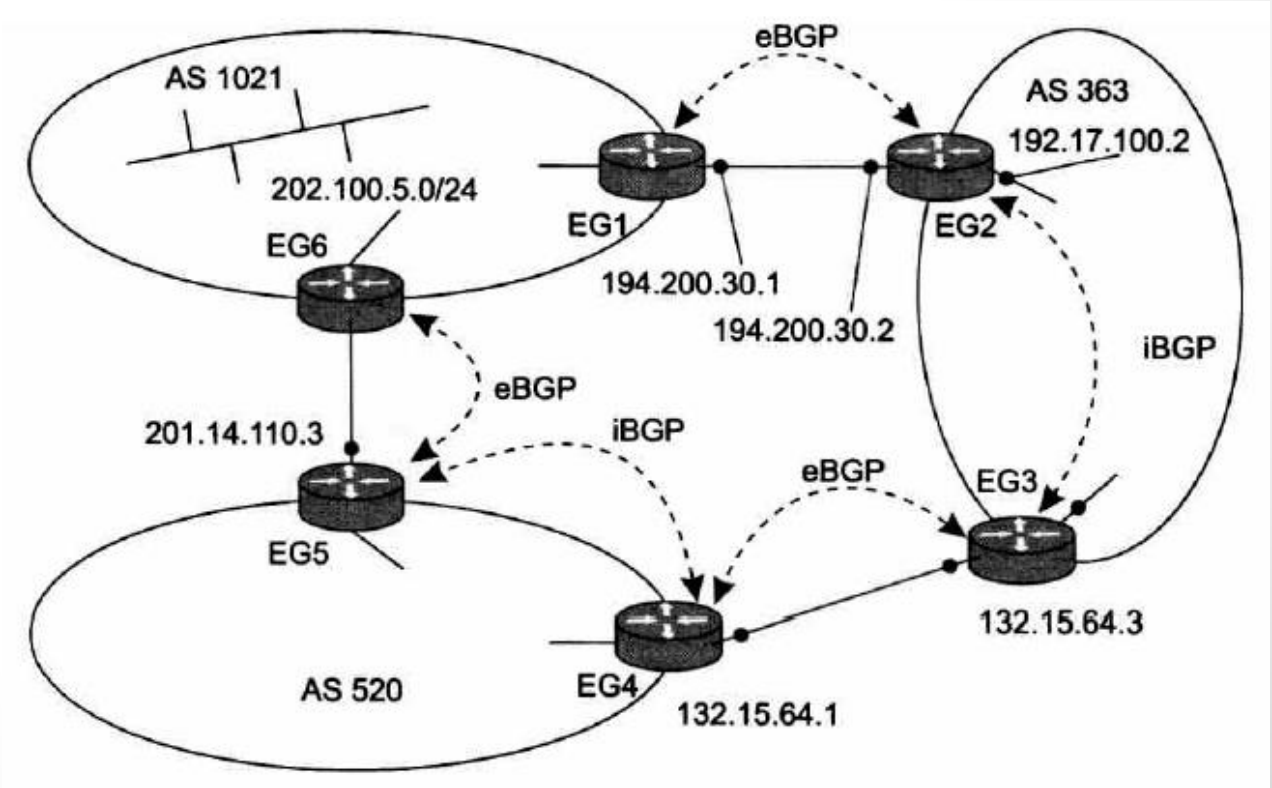
Внутри каждой автономной системы может применяться **любой** из существующих протоколов маршрутизации, в то время как между автономными системами всегда применяется **один и тот же** протокол.



Протокол BGP

В каждой из трех автономных систем (**AS 1021**, **AS 363** и **AS 520**) имеется несколько маршрутизаторов, исполняющих роль внешних шлюзов.

На каждом из них работает протокол **BGP**, с помощью которого они общаются между собой.



Протокол BGP

Маршрутизатор взаимодействует с другими маршрутизаторами по протоколу BGP **только в том случае**, если администратор **явно** указывает при конфигурировании, что эти маршрутизаторы являются его **соседями** – такой способ взаимодействия удобен в ситуации, когда маршрутизаторы, обменивающиеся маршрутной информацией, принадлежат **разным поставщикам услуг**.

Для установления сеанса с указанными соседями маршрутизаторы BGP используют протокол **TCP** (**порт 179**).

Протокол BGP

Основным сообщением протокола BGP является сообщение **UPDATE** (**обновить**), с помощью которого маршрутизатор сообщает маршрутизатору соседней автономной системы о достижимости сетей, относящихся к его собственной автономной системе.

Это **триггерное объявление**, которое посылается соседу только тогда, когда в автономной системе что-нибудь резко меняется: появляются новые сети или новые пути к сетям, а также если исчезают существовавшие сети или пути.

Протокол BGP

Передача информации о сетях между АС:

1. Маршрутизатор **EG1** объявляет маршрутизатору **EG2** о том, что в **AS 1021** появилась новая сеть **202.100.5.0/24** следующим сообщением:

AS 1021; 194.200.30.1; 202.100.5.0/24.

2. Маршрутизатор **EG2**, получив сообщение **UPDATE**, запоминает в своей таблице маршрутизации информацию о сети **202.100.5.0/24** вместе с адресом следующего маршрутизатора **194.200.30.1** и отметкой о том, что эта информация была получена по протоколу **BGP**.

Маршрутизатор **EG2** обменивается маршрутной информацией с внутренними шлюзами системы **AS 363** по какому-либо протоколу группы **IGP** (RIP, EIGRP, OSPF).

Протокол BGP

Передача информации о сетях между АС (продолжение):

3. Все внутренние шлюзы **AS 363** узнают о существовании сети **202.100.5.0/24**. В качестве адреса следующего маршрутизатора маршрутизатор **EG2** начнет теперь объявлять адрес собственного внутреннего интерфейса **192.17.100.2**.

4. Маршрутизаторы **EG2** и **EG3** также устанавливают между собой BGP-сеанс, хотя они и принадлежат одной и той же автономной системе.

Такая реализация протокола BGP называется **внутренней версией BGP** (**Interior BGP**, **iBGP),** в отличие от основной, **внешней версии** (**Exterior BGP**, **eBGP).**

Протокол BGP

Передача информации о сетях между АС (продолжение):

5. Маршрутизатор **EG3** получает нужную информацию от маршрутизатора **EG2** и передает ее внешнему соседу – маршрутизатору **EG4**.

При формировании нового сообщения **UPDATE** маршрутизатор **EG3** трансформирует сообщение, полученное от маршрутизатора **EG2**, добавляя в список автономных систем собственную автономную систему **AS 520**, а полученный адрес следующего маршрутизатора заменяет адресом собственного интерфейса:

AS 363, AS 1021; 132.15.64.3; 202.100.5.0/24.

Протокол BGP

Передача информации о сетях между АС (продолжение):

6. Номера автономных систем позволяют исключить зацикливание сообщений **UPDATE**.

Когда маршрутизатор **EG5** передаст сообщение о сети **202.100.5.0/24** маршрутизатору EG6, то последний не станет его использовать, так как оно будет иметь вид:

AS 520, AS 363, AS 1021; 201.14.110.3; 202.100.5.0/24.

Так как в списке автономных систем **уже есть номер собственной автономной системы**, очевидно, что сообщение **зациклилось**.

Протокол ICMP

Протокол межсетевых управляющих сообщений (**Internet Control Message Protocol**, **ICMP**) является вспомогательным протоколом, использующимся для диагностики и мониторинга сети.

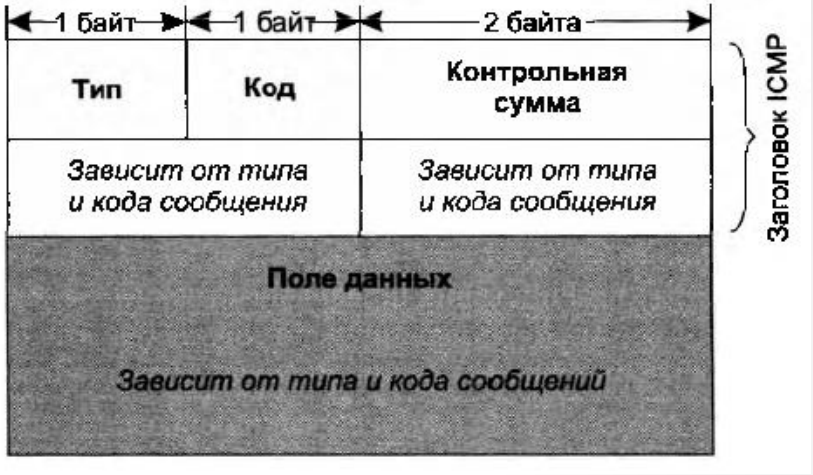
Протокол **ICMP** призван **компенсировать ненадежность** протокола **IP**.

Он не предназначен для исправления возникших при передаче пакета проблем (если пакет потерян, ICMP не может послать его заново).

Задача ICMP в том, что он является **средством оповещения** отправителя о «несчастных случаях», произошедших с его пакетами.

Протокол ICMP

Заголовок ICMP-сообщения состоит из 8 байт:



- **тип** (1 байт) – числовой идентификатор типа сообщения;
- **код** (1 байт) – числовой идентификатор, более тонко дифференцирующий тип ошибки;
- **контрольная сумма** (2 байта) – подсчитывается для всего ICMP-сообщения.

Содержимое оставшихся четырех байтов в заголовке и поле данных зависят от значений полей типа и кода.

Протокол ICMP

Таблица типов ICMP-сообщений

Значение в поле «Тип»	Тип сообщения
0	Эхо-ответ
3	Узел назначения недостижим
4	Подавление источника
5	Перенаправление маршрута
8	Эхо-запрос
11	Истечение времени диаграммы
12	Проблема с параметрами пакета
13	Запрос отметки времени
14	Ответ отметки времени
17	Запрос маски
18	Ответ маски

Таблица кодов причин ошибок 3

Код	Причина
0	Сеть недостижима
1	Узел недостижим
2	Протокол недостижим
3	Порт недостижим
4	Ошибка фрагментации
5	Ошибка в маршруте источника
6	Сеть назначения неизвестна
7	Узел назначения неизвестен
8	Узел-источник изолирован
9	Административный запрет
	. . . . .

? сообщение-запрос

i сообщение-ответ

√ сообщение-ошибка

Протокол ICMP

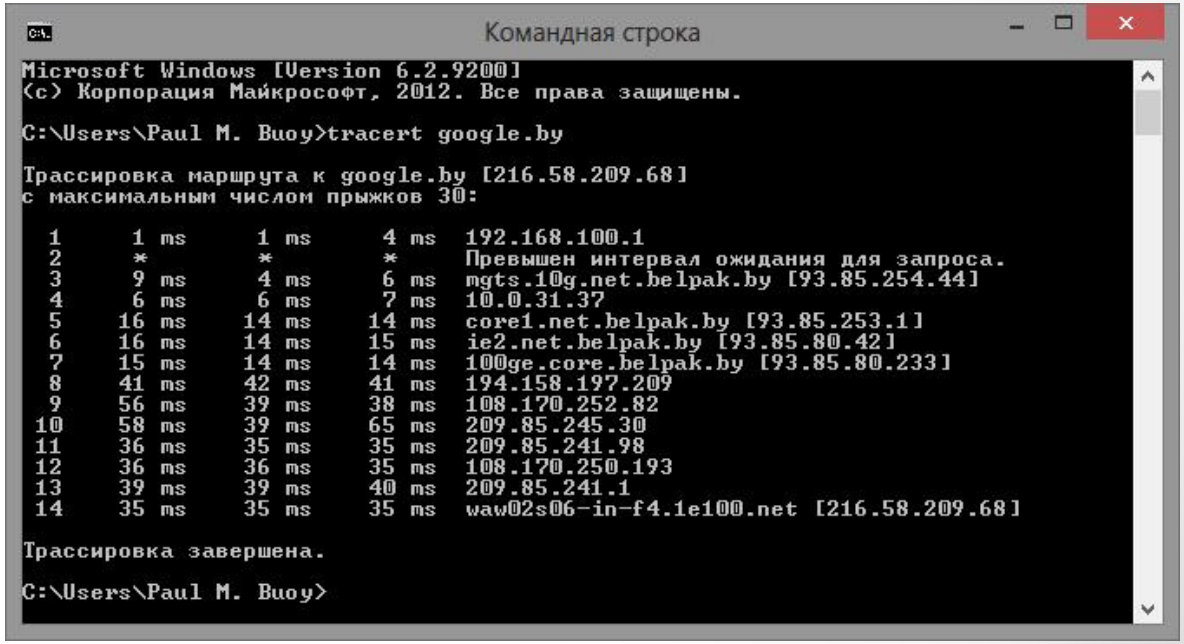
ICMP-сообщения можно разделить на две группы:

- **сообщения об ошибках** – конкретизируются уточняющим кодом;
- **сообщения запрос-ответ** – отправитель сообщения-запроса всегда рассчитывает на получение соответствующего сообщения-ответа.

Протокол ICMP

Утилита **traceroute** (**tracert**) позволяет проследить **маршрут до удаленного хоста**, определить **среднее время оборота** (**RTT**), **IP-адрес** и в некоторых случаях **доменное имя** каждого промежуточного маршрутизатора.

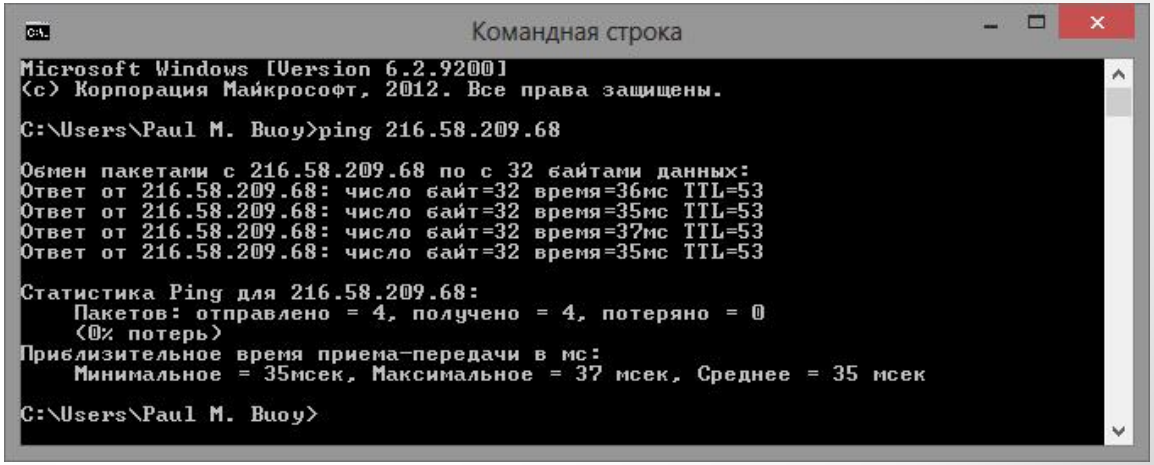
Такая информация помогает найти маршрутизатор, на котором оборвался путь пакета к удаленному хосту.



Протокол ICMP

Утилита **ping** (**эхо-протокол)** представляет собой очень простое средство мониторинга сети.

Компьютер или маршрутизатор посылает по составной сети ICMP-сообщение эхо-запроса, указывая в нем IP-адрес узла, достижимость которого нужно проверить, а узел, получивший эхо-запрос, формирует и отправляет эхо-ответ отправителю запроса.



Успешная доставка означает **нормальное функционирование** всей транспортной системы составной сети.

Переход на IPv6

В результате ряда проблем с IPv4 сообщество Интернета после достаточно долгого обсуждения решило подвергнуть этот протокол серьезной переработке, выбрав в качестве основных целей модернизации:

- **создание масштабируемой схемы адресации**;
- **сокращение объема работы, выполняемой маршрутизаторами**;
- **предоставление гарантий качества транспортных услуг**;
- **обеспечение защиты данных, передаваемых по сети**.

Переход на IPv6

При разработке IPv6 была предусмотрена возможность **плавного перехода к новой версии**, когда довольно значительное время будут сосуществовать островки Интернета, работающие по протоколу IPv6, и остальная часть Интернета, работающая по протоколу IPv4.

Существует несколько подходов к организации взаимодействия узлов, использующих разные стеки TCP/IP:

- **трансляция протоколов**;
- **мультиплексирование стеков протоколов**;
- **инкапсуляция, или туннелирование.**

Переход на IPv6

Трансляция протоколов реализуется **шлюзами**, которые устанавливаются на границах сетей, использующих разные версии протокола IP.

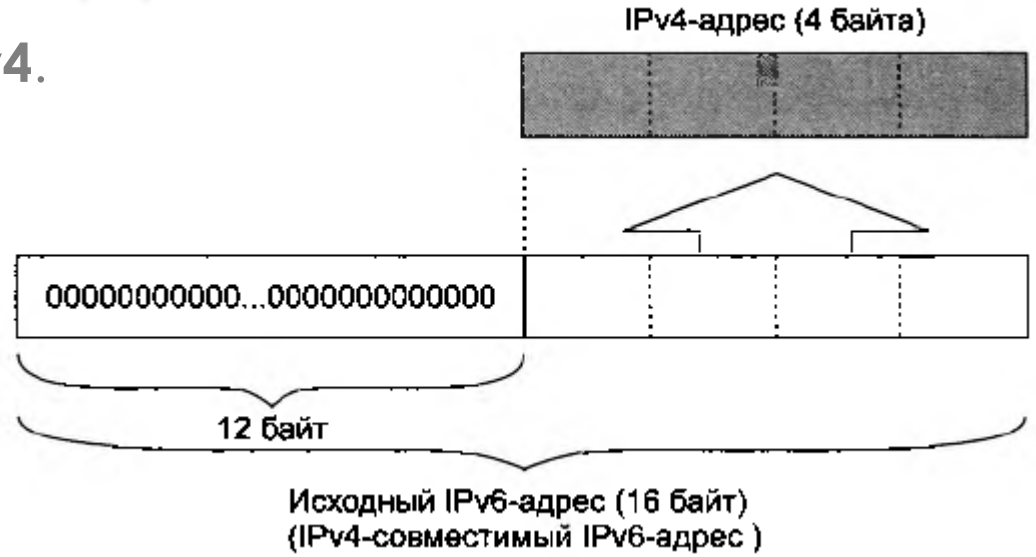
Согласование двух версий протокола IP происходит путем преобразования пакетов IPv4 в IPv6 и наоборот.

Для упрощения преобразования адресов между версиями предлагается использовать специальные подтипы адресов:

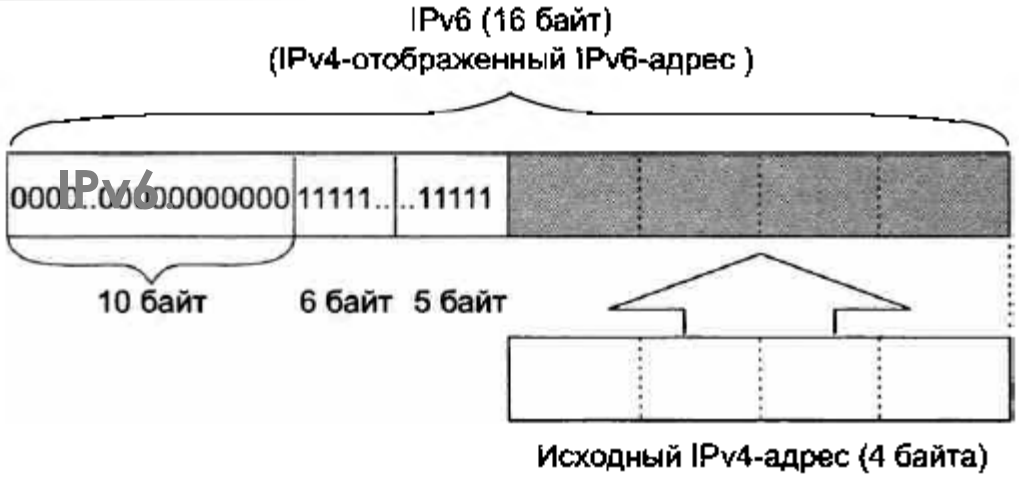
- **IPv4-совместимый IPv6-адрес** (преобразование **IPv6** в **IPv4**);
- **IPv4-отображенный IPv6-адрес** (преобразование **IPv4** в **IPv6**).

Переход на IPv6

Преобразование **IPv6** в **IPv4**.



Преобразование **IPv4** в **IPv6**



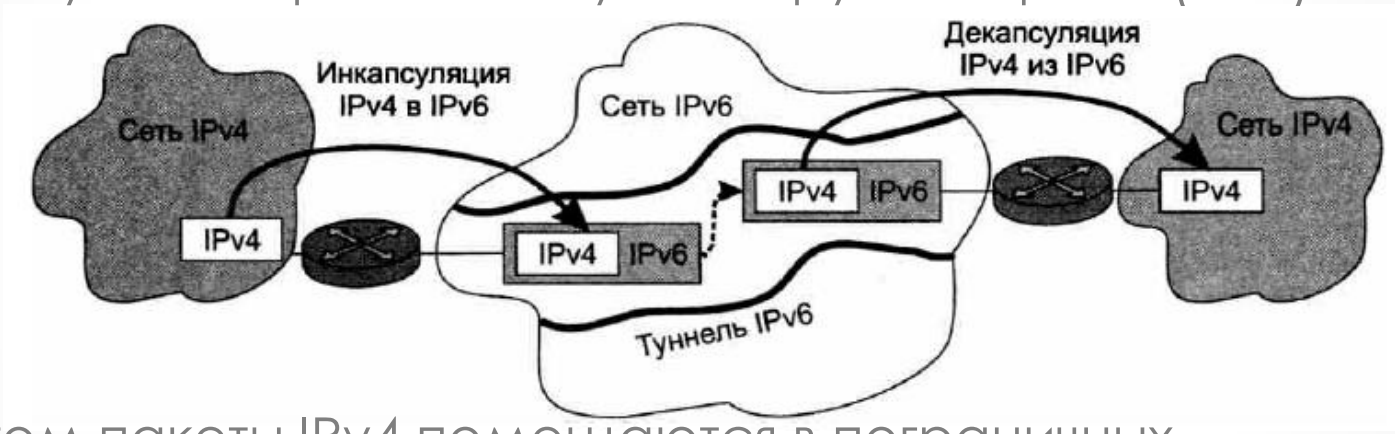
Переход на IPv6

Мультиплексирование стеков протоколов означает установку на взаимодействующих хостах сети и на разделяющих эти хосты маршрутизаторах обеих версий протокола **IP.**

В том случае, когда **IPv6-хост** отправляет сообщение **IPv6-хосту**, он использует стек **IPv6,** а если тот же хост взаимодействует с **IPv4-хостом** – стек **IPv4.**

Переход на IPv6

Инкапсуляция может быть применена, когда две сети одной версии протокола (IPv4), необходимо соединить через транзитную сеть, работающую по другой версии (IPv6).



При этом пакеты IPv4 помещаются в пограничных устройствах в пакеты IPv6 и переносятся через «**туннель**», проложенный в IPv6-сети.

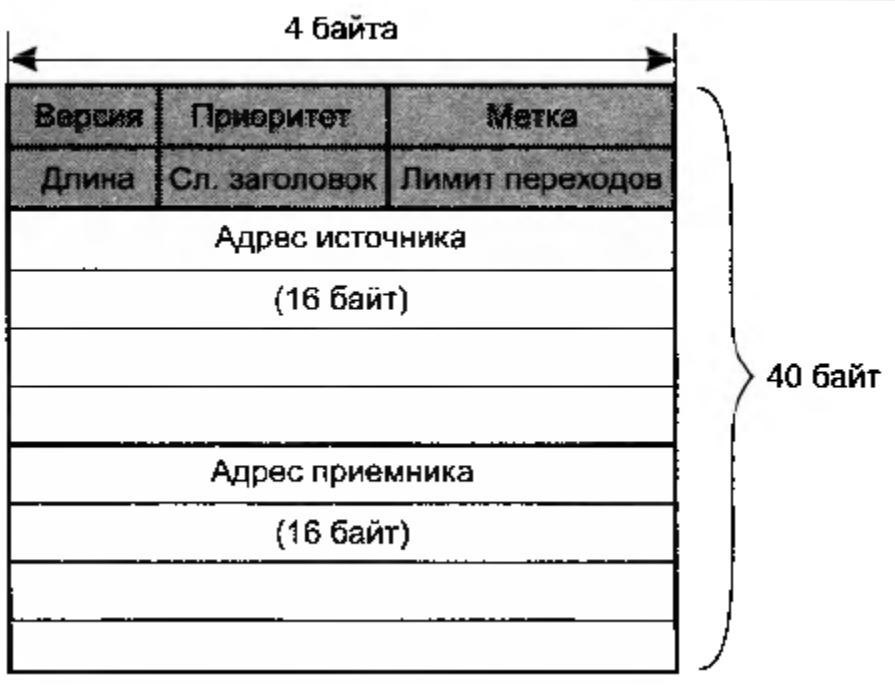
Аналогичным образом метод туннелирования может использоваться для переноса пакетов IPv6 через сеть маршрутизаторов IPv4.

Структура заголовка IPv6

Одной из основных целей изменения формата заголовка протокола IPv6 было **сокращение накладных расходов**, то есть **уменьшение объема служебной информации**, **передаваемой с каждым паке**

Основной заголовок имеет фиксированную длину в **40 байт**.

[Смотри заголовок IPv4]



Структура заголовка IPv6

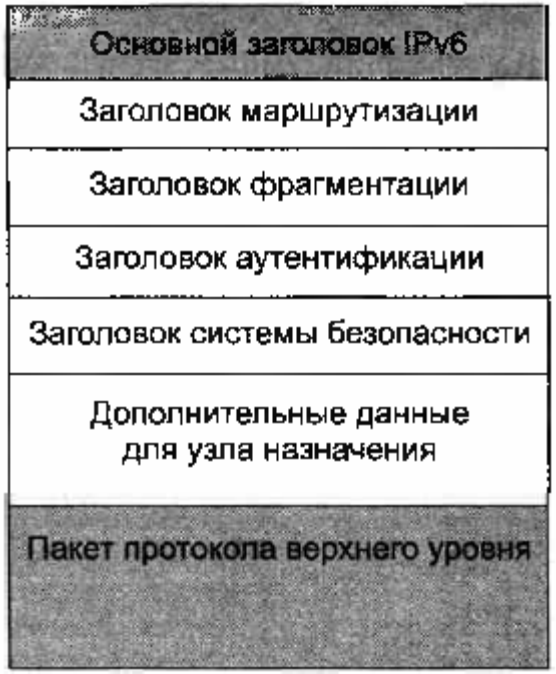
Типы дополнительных заголовков:

- **заголовок маршрутизации** – указание полного маршрута при маршрутизации от источника;
- **заголовок фрагментации** – информация, относящаяся к фрагментации IP-пакета;
- **заголовок аутентификации** – информация, необходимая для аутентификации и обеспечения целостности пакетов;
- **заголовок системы безопасности** – информация, необходимая для шифрования и дешифрирования данных;
- **специальные параметры** – параметры, необходимые для последовательной обработки пакетов на каждом маршрутизаторе;
- **параметры получателя** – дополнительная информация.

Структура заголовка IPv6

Поскольку для маршрутизации пакета обязательным является лишь основной заголовок (почти все дополнительные заголовки обрабатываются только в конечных узлах), **это снижает нагрузку на маршрутизаторы**.

В то же время возможность использования большого количества дополнительных параметров **расширяет функциональность протокола IP** и делает его **открытым для внедрения новых механизмов**.



Структура заголовка IPv6

Для того чтобы **повысить производительность маршрутизаторов** в версии IPv6 предпринят ряд мер по освобождению их от некоторых вспомогательных функций:

- **перенесение функций фрагментации с маршрутизаторов на конечные узлы** – конечные узлы обязаны найти минимальное значение MTU вдоль всего пути;
- **агрегирование адресов** – уменьшению размера адресных таблиц маршрутизаторов;
- **широкое использование маршрутизации от источника** – освобождает маршрутизаторы от необходимости просмотра адресных таблиц;
- **отказ от обработки необязательных параметров заголовка**;
- **использование в качестве номера узла его MAC-адреса** – избавляет маршрутизаторы от необходимости применять

Адресация IPv6

IPv6 внесла существенные изменения в систему адресации – прежде всего это коснулось увеличения разрядности адреса: вместо **4 байт** (IPv4) в новой версии под адрес отведено **16 байт**.

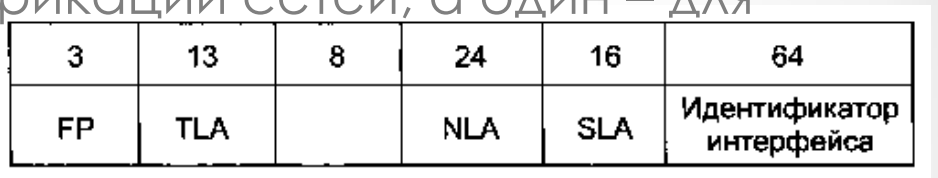
Это дает возможность пронумеровать **ОГРОМНОЕ** количество узлов:

340 282 366 920 938 463 463 374 607 431 762 211 456.

Главной целью изменения системы адресации было не механическое увеличение адресного пространства, а **повышение эффективности работы стека TCP/IP** в целом.

Адресация IPv6

Вместо прежних **двух уровней иерархии адреса** (номер сети и номер узла) в IPv6 определено **четыре уровня**, из которых три служат для идентификации сетей, а один – для идентификации узлов сети:



3	13	8	24	16	64
FP	TLA		NLA	SLA	Идентификатор интерфейса

- **префикс формата** (**Format Prefix**, **FP**) – определяет тип адреса;
- **поле TLA** (**Top-Level Aggregation**) предназначено для идентификации сетей самых крупных поставщиков услуг;
- **поле NLA** (**Next-Level Aggregation**) предназначено для нумерации сетей средних и мелких поставщиков услуг;
- **поле SLA** (**Site-Level Aggregation**) предназначено для адресации подсетей отдельного абонента (одной корпоративной сети);
- **идентификатор интерфейса** является аналогом номера узла в IPv4.

Адресация IPv6

Идентификатор интерфейса имеет длину **64 бита**, что позволяет поместить туда MAC-адрес (48 бит), адрес конечного узла ATM (48 бит) или номер виртуального соединения ATM (до 28 бит), а также, вероятно, даст возможность использовать локальные адреса технологий, которые могут появиться в будущем.

Такой подход **делает ненужным протокол ARP**, поскольку процедура отображения IP-адреса на локальный адрес становится тривиальной – она сводится к простому отбрасыванию старшей части адреса.

Адресация IPv6

В новой версии не поддерживаются классы адресов (A, B, C, D, E), но широко используется технология CIDR, что вместе с усовершенствованной системой групповой адресации и введением адресов нового типа IPv6 позволяет **сократить затраты на маршрутизацию**.

При изобилии сетей, которое предоставляется клиенту в IPv6, совершенно **теряет смысл операция использования масок для разделения сетей на подсети**, в то время как обратная процедура – **объединение подсетей – приобретает особое значение**.

Агрегирование адресов является основным способом **эффективного расходования адресного пространства** в новой версии протокола IP.

Адресация IPv6

Разработчики стандарта предложили использовать вместо десятичной **шестнадцатеричную** форму записи IP-адреса.

Каждые четыре шестнадцатеричные цифры отделяются друг от друга двоеточием:

FEDC:0A98:0:0:0:0:7654:3210

или

FEDC:0A98::7654:3210

Для сетей, поддерживающих обе версии протокола (IPv4 и IPv6), разрешается задействовать для младших четырех байт традиционную для IPv4 десятичную запись:

FEDC:0A98:0:0:0:0:118.84.50.16

Адресация IPv6

В новой версии IPv6 предусмотрено три основных типа адресов:

- **индивидуальный адрес** (**unicast**) является уникальным идентификатором отдельного интерфейса конечного узла или маршрутизатора;
- **групповой адрес** (**multicast**) аналогичен по назначению групповому адресу IPv4 – он идентифицирует группу интерфейсов, относящихся, как правило, к разным узлам;
- **адрес произвольной рассылки** (**anycast**) – это новый тип IP-адреса, определяющий группу интерфейсов (пакет, в поле адреса назначения которого стоит адрес произвольной рассылки, доставляется **одному** из интерфейсов группы, как правило, «ближайшему», в соответствии с метрикой, используемой протоколами маршрутизации).

Адресация IPv6

Тип адреса определяется значением нескольких старших битов адреса, которые названы **префиксом формата.**

Индивидуальные адреса делятся на несколько подтипов:

- **глобальный индивидуальный адрес** – адрес, к которому можно проложить маршрут по Интернету, он является уникальным по всему миру;
- **локальный адрес канала** – используются для обмена данными с другими устройствами по одному локальному каналу (подсети);

Адресация IPv6

Индивидуальные адреса (продолжение):

- **логический интерфейс loopback** (::1/128) – используется узлом для отправки пакета самому себе и не может быть назначен физическому интерфейсу;
- **неопределённый адрес** (::/128) – может быть назначен интерфейсу и используется только в качестве адреса источника в IPv6-пакете;
- **уникальный локальный адрес** (от FC00::/7 до FDFF::/7) – используются для локальной адресации в пределах узла или между ограниченным количеством узлов, их не следует маршрутизировать в глобальном протоколе IPv6.

Транспортный уровень

Транспортный уровень стека TCP/IP может предоставлять вышележащему уровню два типа сервиса:

- **гарантированную доставку** обеспечивает **протокол управления передачей (Transmission Control Protocol, TCP);**

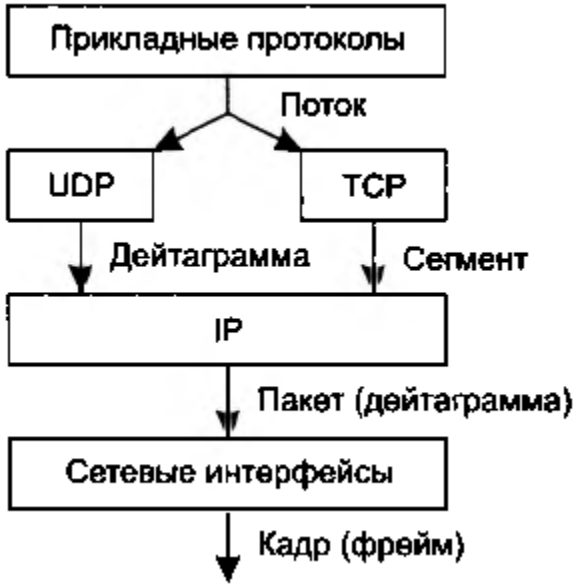
- **доставку по возможности**, или с максимальными усилиями, обеспечивает **протокол пользовательских дейтаграмм (User Datagram Protocol, UDP).**

Транспортный уровень

Потоком данных, информационным потоком, или просто **потоком,** называют данные, поступающие от приложений на вход протоколов транспортного уровня – **TCP** и **UDP**.

Протокол TCP «нарезает» из потока данных **сегменты.**

Единицу данных протокола UDP часто называют **дейтаграммой,** или **датаграммой.**



Дейтаграмма – это общее название для единиц данных, которыми оперируют протоколы без установления соединений.

Порты

Каждый компьютер может одновременно выполнять **несколько процессов**, даже отдельный прикладной процесс может иметь **несколько точек входа**, выступающих в качестве адресов назначения для пакетов данных.

Реализуемая протоколами **TCP** и **UDP** процедура распределения между прикладными процессами пакетов, поступающих от сетевого уровня, называется **демультиплексированием**.

Обратная задача – **мультиплексирование**.

Порты

Протоколы **TCP** и **UDP** ведут для **каждого** приложения две системные очереди:

- очередь данных, **поступающих к приложению из сети**;
- очередь данных, **отправляемых приложением в сеть**.

Эти системные очереди называются **портами** (входная и выходная очереди одного приложения рассматриваются как один порт).

Для идентификации портов им присваивают **номера**.

Порты

Если процессы представляют собой популярные системные службы (**FTP**, **telnet**, **HTTP**, **TFTP**, **DNS** и т. п.), то за ними **централизовано** закрепляются **стандартные назначенные номера**, называемые также **хорошо известными** (**well-known**) номерами портов в диапазоне от **0** до **1023**.

Порт	Name	Комментарий
1	tcpmux	TCP-порт службы мультиплексора
5	rje	Ввод удалённого задания
7	echo	Служба Echo
9	discard	Служба-пустышка для проверки соединения
11	systat	Служба системного состояния, показывающая подключенные порты
13	daytime	Возвращает запрашивающему узлу дату и время
17	qotd	Выдаёт подключенному узлу фразу дня
18	msp	Протокол отправки сообщений
19	chargen	Служба, генерирующая символы; посылает бесконечный поток символов
20	ftp-data	Порт данных FTP
21	ftp	Порт протокола передачи файлов (File Transfer Protocol, FTP); иногда используется протоколом файловой службы (File Service Protocol, FSP)
22	ssh	Служба безопасной оболочки (Secure SHell, SSH)
23	telnet	Служба Telnet

Порты

Для менее распространенных приложений номера портов назначаются **локально** разработчиками этих приложений или **динамически** операционной системой (она ведет список занятых и свободных номеров портов) из диапазона от **1024** до **65 535** в ответ на поступление запроса от приложения.

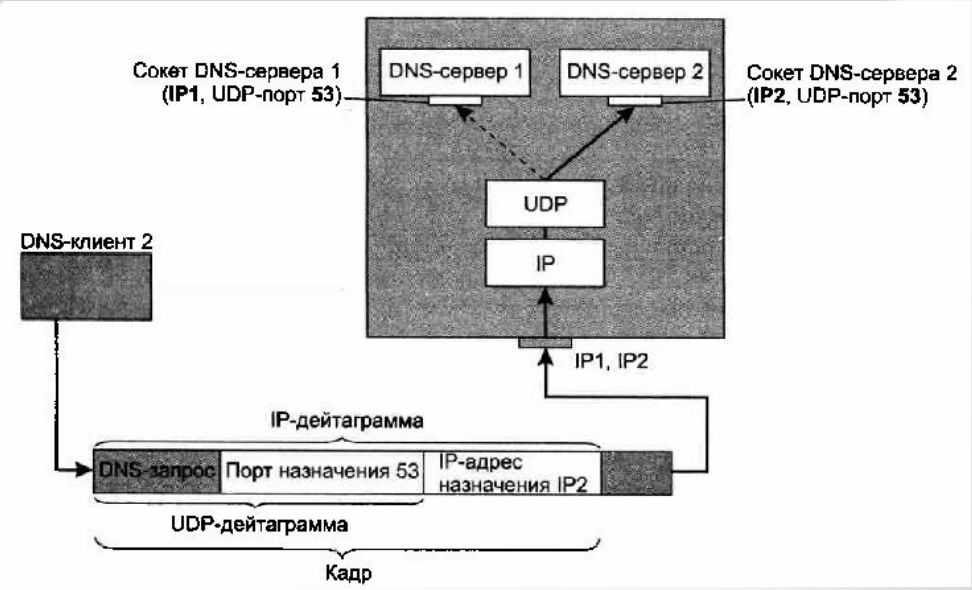
Приложения, которые передают данные на уровень IP по протоколу **UDP**, получают номера, называемые **UDP-портами.** Аналогично, приложениям, обращающимся к протоколу **TCP**, выделяются **TCP-порты.**

Сокеты

Одного номера порта не достаточно, чтобы однозначно определить прикладной процесс в пределах хоста.

Прикладной процесс однозначно определяется в пределах сети и в пределах отдельного компьютера парой (IP-адрес, номер порта), называемой **сокетом** (**socket**).

Сокет, определенный IP-адресом и номером **UDP**-порта, называется **UDP-сокетом,** а IP-адресом и номером **TCP**- порта – **TCP-сокетом.**



Сокеты

Порядок использования сокетов для получения приложениями данных:

1) получение IP- пакета от протокола канального уровня;

2) анализ протоколом IP содержимого заголовка пакета и его отбрасывание;

3) Запоминание IP-адреса назначения из заголовка IP-пакета;

4) передача UDP-дейтаграммы или TCP-сегмента в порт приложения, с использованием IP-адреса для однозначной его идентификации.

Протокол UDP

Протокол пользовательских дейтаграмм (**User Datagram Protocol**, **UDP)** – это протокол транспортного уровня, который обеспечивает доставку по возможности, или с максимальными усилиями**.**

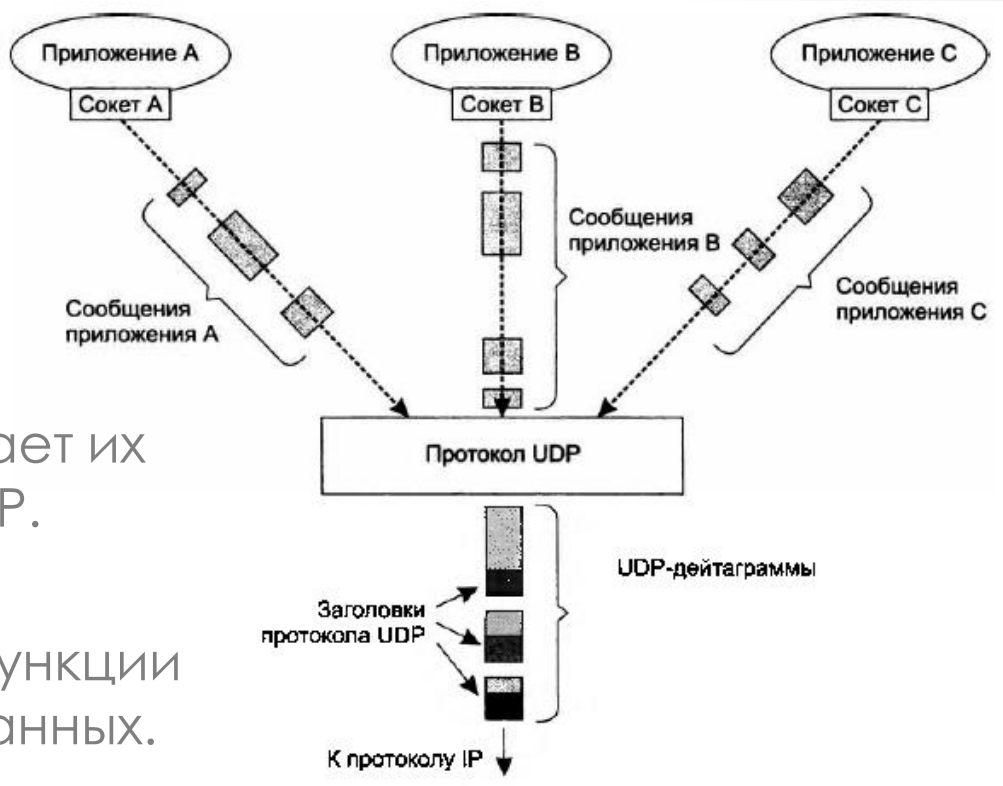
Подобно IP UDP является дейтаграммным протоколом, который **не гарантирует** доставку сообщений адресату.

Протокол UDP

При работе **на хосте-отправителе** данные от приложений поступают протоколу **UDP** через порт в виде сообщений.

Протокол UDP добавляет к каждому отдельному сообщению свой **8-байтный заголовок**, формируя из этих сообщений **UDP-дейтаграммы**, и передает их нижележащему протоколу IP.

В этом и заключаются его функции по **мультиплексированию** данных.



Процесс передачи UDP-дейтаграмм

Каждая дейтаграмма переносит **отдельное пользовательское сообщение**.

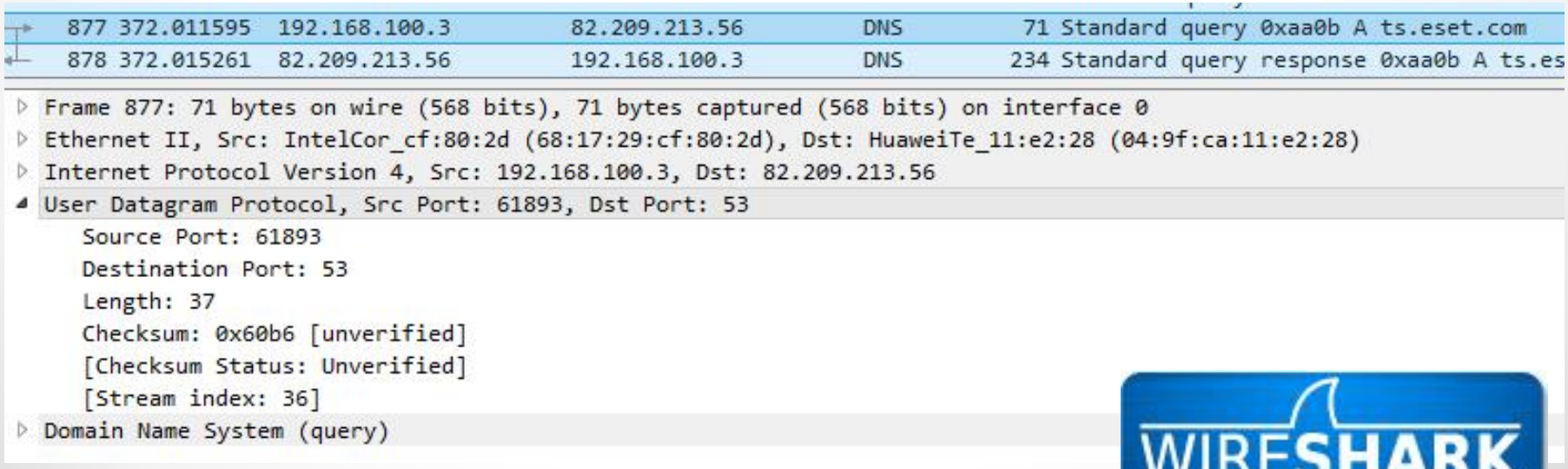
Сообщения могут иметь разную длину, не превышающую длину поля данных протокола IP, которое, в свою очередь, ограничено размером кадра технологии нижнего уровня.

Если буфер UDP переполняется, то сообщение приложения **отбрасывается**.

Протокол UDP

Заголовок UDP состоит из четырех 2-байтных полей:

- **номер UDP-порта отправителя**;
- **номер UDP-порта получателя**;
- **длина дейтаграммы**;
- **контрольная сумма.**



Процесс передачи UDP-дейтаграмм

Функции **UDP** сводятся к простой передаче данных между прикладным и сетевым уровнями, а также к примитивному контролю искажений в передаваемых данных.

При контроле искажений протокол UDP только **диагностирует**, **но не исправляет ошибку**.

Если контрольная сумма показывает, что в поле данных UDP-дейтаграммы произошла ошибка, протокол UDP **отбрасывает поврежденную дейтаграмму**.

Протокол UDP

Работая **на хосте-получателе**, протокол UDP принимает от протокола IP извлеченные из пакетов UDP-дейтаграммы.

Полученные из IP-заголовка **IP-адрес назначения** и из UDP-заголовка **номер порта** используются для формирования UDP-сокета, однозначно идентифицирующего приложение, которому направлены данные.

Протокол UDP освобождает дейтаграмму от UDP-заголовка и полученное в результате сообщение передает приложению на соответствующий UDP-сокет.

Так протокол UDP выполняет **демультиплексирование** на основе сокетов.

Протокол UDP

Ряд приложений, использующих протокол UDP на транспортном уровне:

Приложение	Порт
DNS	53
BOOTP	клиент 68, сервер 67
DHCPv4	клиент 68, сервер 67
DHCPv6	клиент 546, сервер 547
TFTP	69
HTTPS	443
NTP	123
Syslog	514
SNMP	162

Протокол TCP

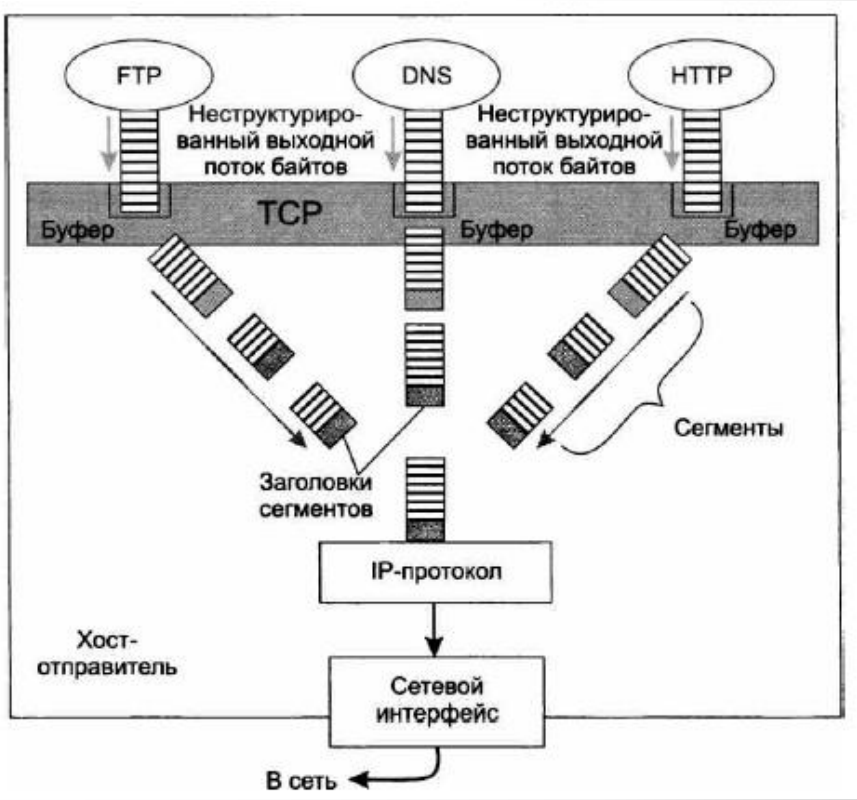
Протокол управления передачей (**Transmission Control Protocol**, **TCP)** предназначен для передачи данных между приложениями и основан на **логическом соединении**, что позволяет ему обеспечивать **гарантированную доставку** данных, используя в качестве инструмента ненадежный дейтаграммный сервис протокола IP.

Протокол TCP

При работе **на хосте-отправителе** протокол TCP рассматривает информацию, поступающую к нему от прикладных процессов, как **неструктурированный поток байтов**.

Поступающие данные буферизуются средствами TCP.

Для передачи на сетевой уровень из буфера «вырезается» некоторая непрерывная часть данных, которая называется **сегментом** и снабжается заголовком.



Протокол TCP

В отличие от протокола UDP, который создает свои дейтаграммы на основе **логически обособленных единиц данных – сообщений**, генерируемых приложениями, протокол TCP делит поток данных на сегменты **без учета их смысла или внутренней структуры**.

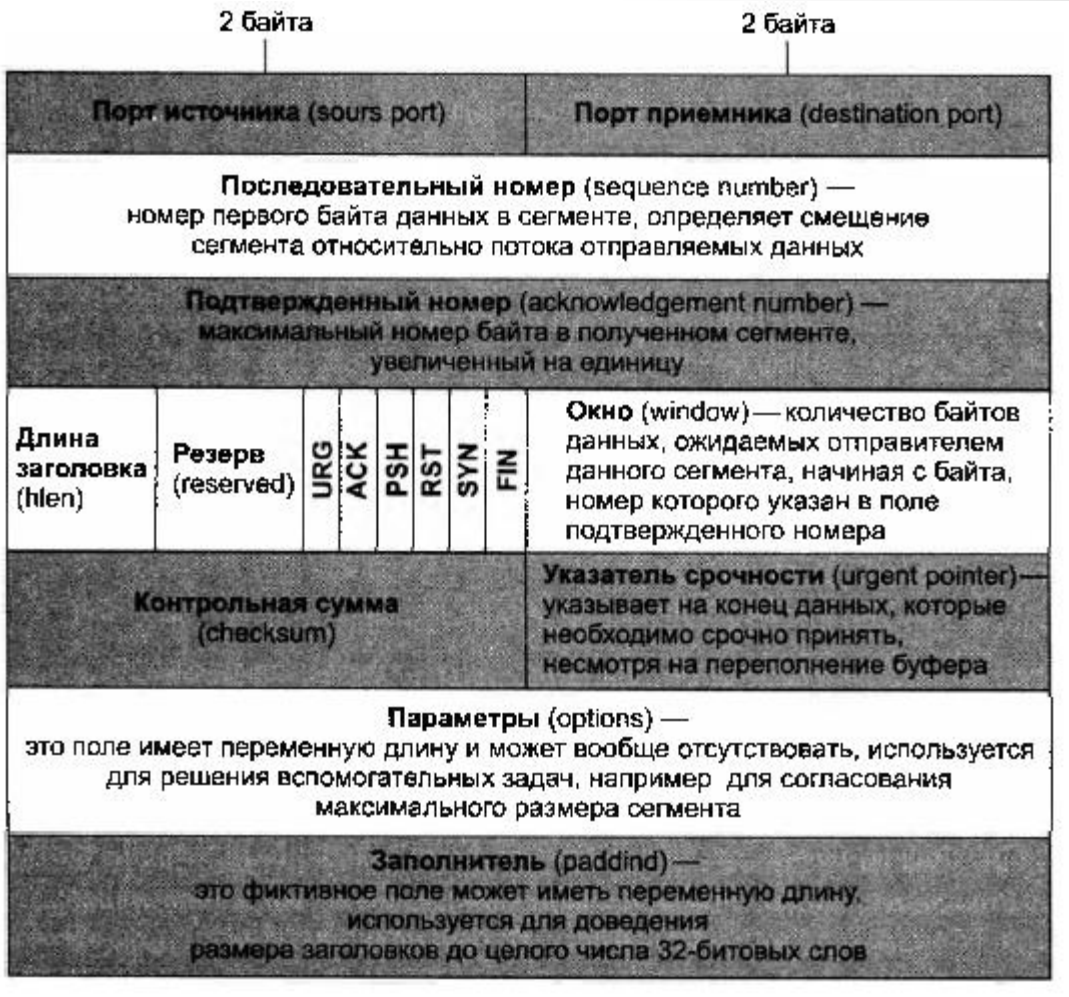
Протокол TCP

Ряд приложений, использующих протокол TCP на транспортном уровне:

Приложение	Порт
DNS	53
SMTP	25
POP	110
IMAP	143
FTP	20, 21
HTTP	80, 8080
HTTPS	443
SSH	22
Telnet	23

TCP-сегмент

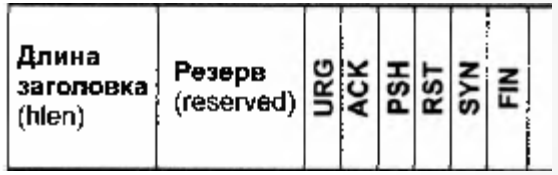
Заголовок TCP-сегмента содержит значительно больше полей, чем заголовок UDP, что отражает **более развитые возможности протокола TCP**.



TCP-сегмент

Однобитные поля, называемых **флагами,** или **кодовыми битами** (**code bits**) расположены сразу за резервным полем и содержат служебную информацию о типе сегмента.

Положительное значение сигнализируется установкой этих битов в единицу:



URG – срочное сообщение;

ACK – квитанция на принятый сегмент;

PSH – запрос на отправку сообщения 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.

TCP-сегмент

```
  870 371.976240  91.190.216.58    192.168.100.3   TCP      54 12350 → 52094 [ACK] Seq=68 Ack=226 Win=8192 Len=0
  871 371.981638  157.55.130.158   192.168.100.3   TCP     134 40031 → 52095 [PSH, ACK] Seq=1 Ack=76 Win=14848 Len=80
  872 371.981971  91.190.216.58    192.168.100.3   TCP     357 12350 → 52094 [PSH, ACK] Seq=68 Ack=226 Win=8192 Len=303

▷ Frame 871: 134 bytes on wire (1072 bits), 134 bytes captured (1072 bits) on interface 0
▷ Ethernet II, Src: HuaweiTe_11:e2:28 (04:9f:ca:11:e2:28), Dst: IntelCor_cf:80:2d (68:17:29:cf:80:2d)
▷ Internet Protocol Version 4, Src: 157.55.130.158, Dst: 192.168.100.3
◢ Transmission Control Protocol, Src Port: 40031, Dst Port: 52095, Seq: 1, Ack: 76, Len: 80
    Source Port: 40031
    Destination Port: 52095
    [Stream index: 4]
    [TCP Segment Len: 80]
    Sequence number: 1    (relative sequence number)
    [Next sequence number: 81    (relative sequence number)]
    Acknowledgment number: 76    (relative ack number)
    Header Length: 20 bytes
  ◢ Flags: 0x018 (PSH, ACK)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...1 .... = Acknowledgment: Set
      .... .... 1... = Push: Set
      .... .... .0.. = Reset: Not set
      .... .... ..0. = Syn: Not set
      .... .... ...0 = Fin: Not set
      [TCP Flags: ·······AP···]
    Window size value: 29
    [Calculated window size: 14848]
    [Window size scaling factor: 512]
    Checksum: 0xa76c [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▷ [SEQ/ACK analysis]
▷ Data (80 bytes)
```

TCP-сегмент

```
911 372.165871  91.228.167.146        192.168.100.3         TCP         54 80 → 52097 [FIN, ACK] Seq=380 Ack=2482 Win=31064 Len=0

▷ Frame 911: 54 bytes on wire (432 bits), 54 bytes captured (432 bits) on interface 0
▷ Ethernet II, Src: HuaweiTe_11:e2:28 (04:9f:ca:11:e2:28), Dst: IntelCor_cf:80:2d (68:17:29:cf:80:2d)
▷ Internet Protocol Version 4, Src: 91.228.167.146, Dst: 192.168.100.3
◢ Transmission Control Protocol, Src Port: 80, Dst Port: 52097, Seq: 380, Ack: 2482, Len: 0
    Source Port: 80
    Destination Port: 52097
    [Stream index: 6]
    [TCP Segment Len: 0]
    Sequence number: 380    (relative sequence number)
    Acknowledgment number: 2482    (relative ack number)
    Header Length: 20 bytes
  ◢ Flags: 0x011 (FIN, ACK)
      000. .... .... = Reserved: Not set
      ...0 .... .... = Nonce: Not set
      .... 0... .... = Congestion Window Reduced (CWR): Not set
      .... .0.. .... = ECN-Echo: Not set
      .... ..0. .... = Urgent: Not set
      .... ...1 .... = Acknowledgment: Set
      .... .... 0... = Push: Not set
      .... .... .0.. = Reset: Not set
      .... .... ..0. = Syn: Not set
    ▷ .... .... ...1 = Fin: Set
      [TCP Flags: ·····A···F]
    Window size value: 31064
    [Calculated window size: 31064]
    [Window size scaling factor: -2 (no window scaling used)]
    Checksum: 0x916d [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
  ▷ [SEQ/ACK analysis]
```

Логические соединения

Основным отличием TCP от **UDP** является то, что на протокол TCP возложена дополнительная задача – **обеспечить надежную доставку сообщений**, используя в качестве основы **ненадежный дейтаграммный протокол IP**.

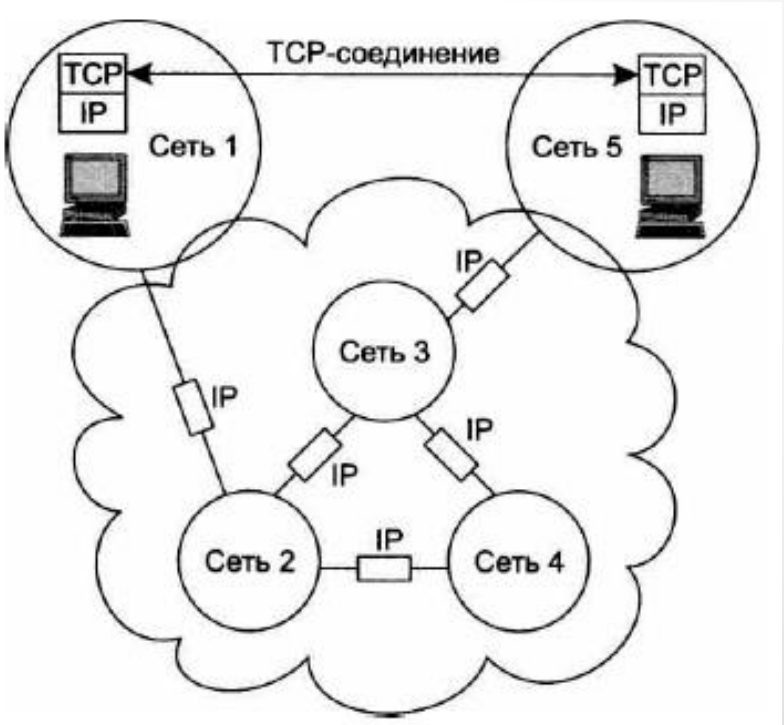
Для решения этой задачи протокол TCP использует метод продвижения данных с установлением **логического соединения**.

Логическое соединение дает возможность участникам обмена следить за тем, чтобы данные **не были потеряны**, **искажены** или **продублированы**, а также чтобы они **пришли к получателю в том порядке**, **в котором были отправлены**.

Логические соединения

Протокол TCP устанавливает логические соединения между **прикладными процессами**, причем в каждом соединении участвуют только **два** процесса.

TCP-соединение является **дуплексным**, то есть каждый из участников этого соединения может одновременно получать и отправлять данные.



Логическое TCP-соединение **однозначно** идентифицируется парой сокетов, определенных для этого соединения двумя взаимодействующими процессами.

Логические соединения

При **установлении логического соединения** модули TCP договариваются между собой о **параметрах процедуры обмена данными**.

В протоколе TCP каждая сторона соединения посылает противоположной стороне следующие параметры:

- максимальный размер сегмента, который она готова принимать;
- максимальный объем данных (количество сегментов), которые она разрешает другой стороне передавать в свою сторону, даже если та еще не получила квитанцию на предыдущую порцию данных (размер окна);
- начальный порядковый номер байта, с которого она начинает отсчет потока данных в рамках данного соединения.

Логические соединения

Соединение устанавливается по инициативе клиентской части приложения.

Клиент – это модуль, предназначенный для формирования и передачи сообщений-запросов к ресурсам удаленного компьютера от разных приложений с последующим приемом результатов из сети и передачей их соответствующим приложениям.

1) При необходимости выполнить обмен данными с серверной частью приложение-клиент обращается к нижележащему протоколу TCP, который в ответ на это обращение посылает **сегмент-запрос** (в запросе содержится флаг **SYN**, **установленный в 1**) на установление соединения протоколу TCP, работающему на стороне сервера.

Логические соединения

2) Получив запрос, модуль TCP на стороне сервера пытается создать «инфраструктуру» для обслуживания нового клиента.

Сервер – это модуль, который постоянно ожидает прихода из сети запросов от клиентов и, приняв запрос, пытается его обслужить.

Он обращается к операционной системе с просьбой о выделении определенных системных ресурсов для организации буферов, таймеров, счетчиков (они закрепляются за соединением с момента создания и до момента разрыва).

Если на стороне сервера все необходимые ресурсы были получены и все необходимые действия выполнены, то модуль TCP посылает клиенту **сегмент** с флагами **ACK** и **SYN**.

Логические соединения

3) В ответ клиент посылает **сегмент** с флагом **ACK** и переходит в состояние установленного логического соединения (состояние ESTABLISHED).

Когда сервер получает флаг **ACK**, он также переходит в состояние ESTABLISHED.

На этом процедура установления соединения заканчивается, и стороны могут переходить к обмену данными.

Логические соединения

Соединение может быть **разорвано** в **любой момент** по инициативе **любой стороны**.

Для этого **клиент** и **сервер** должны обменяться сегментами **FIN** и **ACK**.

Соединение считается закрытым по прошествии некоторого времени, в течение которого сторона-инициатор убеждается, что ее завершающий сигнал **ACK** дошел нормально и не вызвал никаких «аварийных» сообщений со стороны сервера.

Методы квитирования

Передача с квитированием – это один из наиболее естественных приемов, используемых для организации надежного обмена данными.

Отправитель **отсылает данные**, а получатель **подтверждает** их получение **квитанциями**.

Если отправитель вовремя не получает квитанции на переданные данные, то он передает их **повторно**, выполняя **запрос повторной передачи** (**Automatic Repeat reQuest**, **ARQ**).

Методы квитирования

Все многообразие **методов квитирования** может быть разделено на два класса:

- методы **простоя источника** (**Stop-and-Wait**);
- методы **скользящего окна**:
  - методы, использующие **окно передачи** – метод **передачи с возвращением на N пакетов** (**Go-Back-N**);
  - методы, использующие **окно передачи** и **окно приема** – метод **передачи с выборочным повторением** (**Selective Repeat**).

Методы квитирования

Общие черты, присущие всем методам квитирования:

- отправитель и получатель, работающие **асинхронно**, осуществляют передачу пакетов по **ненадежной** линии связи, в которой возможны **искажения**, большие **задержки** и **потери пакетов**;
- отправитель принимает данные от **протокола верхнего уровня** (приложения), получатель передает полученные данные на **верхний уровень** (приложению);
- получатель располагает механизмом определения искаженных пакетов (по контрольной сумме);
- после успешного получения пакета получатель посылает отправителю **квитанции** (**acknowledgment**, **ACK**);
- для отслеживания задержек пакетов используется **таймер**, тайм-аут которого устанавливается равным предельному времени ожидания квитанции.

Метод передачи данных с простоем источника

В методе **простоя источника** отправитель передает последовательность **ненумерованных** пакетов.

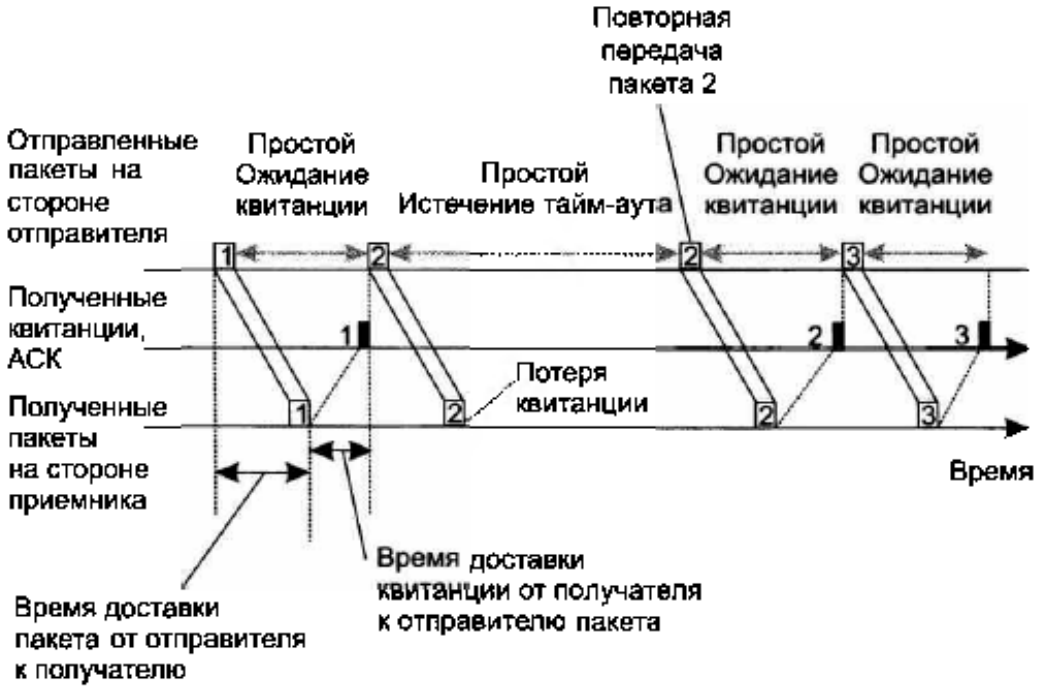
Метод требует, чтобы отправитель дожидался от получателя квитанции и только **после этого** посылал следующий пакет.

С переданным пакетом отправитель связывает **таймер**.

Если в течение тайм-аута квитанция не пришла, то пакет (или квитанция на него) считается утерянным или искаженным и его передача повторяется.

Метод передачи данных с простоем источника

Принимающая сторона должна уметь распознавать дублированные пакеты и отбрасывать их.



Отправитель также должен иметь возможность распознавать дубликаты квитанций.

Метод передачи данных с простоем источника

В заголовок пакета включается **специальный бит** и устанавливается отправителем так, что нуль и единица в качестве его значения чередуются в пределах всей последовательности отправляемых пакетов.

Приемник проверяет значение бита: если у двух последовательно пришедших пакетов значения данного бита равны **двум единицам** или **двум нулям** то эти пакеты считаются **дубликатами**.

Аналогично поступает отправитель с квитанциями.

Метод передачи данных с простоем источника

Такой способ распознавания дубликатов **не является абсолютно надежным**, поскольку он основывается на предположении, что **вероятность прихода в приемник дубликатов друг за другом достаточно высока**.

В данном методе **коэффициент использования линии связи очень низкий** – основную часть времени передатчик простаивает в ожидании прихода квитанции.

Концепция скользящего окна

Концепция **скользящего окна** (**sliding window**) заключается в том, что для повышения скорости передачи данных отправителю разрешается передать некоторое количество пакетов, **не дожидаясь прихода на эти пакеты квитанций**.

Для **идентификации** пакетам присваиваются **уникальные последовательные номера**, которые размещаются в заголовках пакетов.

Разрядность поля «**номер пакета**» определяет диапазон возможных номеров и когда этот диапазон исчерпывается, нумерация пакетов снова начинается с нуля, поэтому **нельзя абсолютно исключить ситуацию**, **когда в сети существуют пакеты с одинаковыми номерами**.

Концепция скользящего окна

Окно определяется на **последовательности пронумерованных пакетов**.

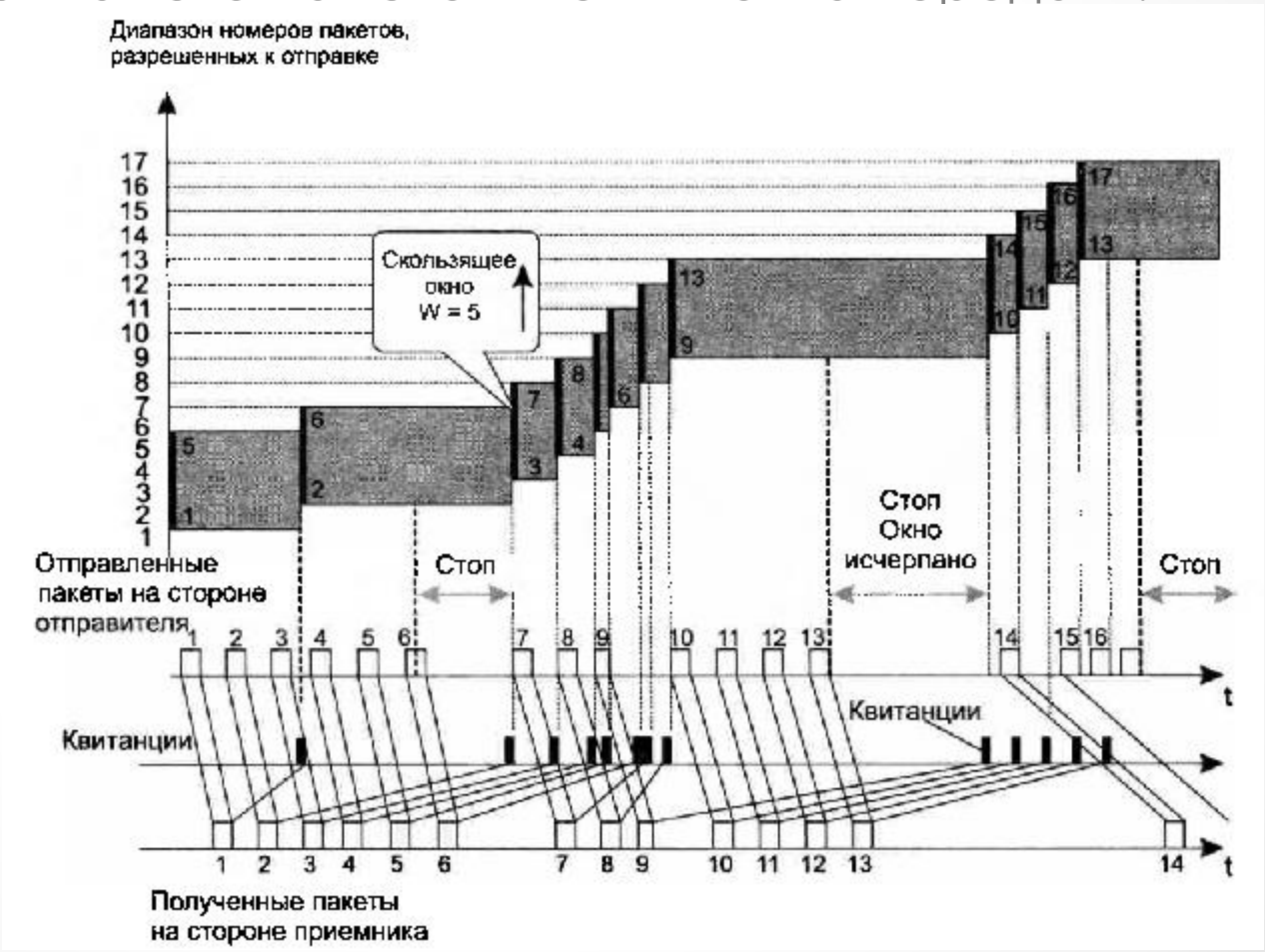
Окно всегда имеет **нижнюю границу**, называемую также **базой окна**, и **верхнюю границу**.

Количество номеров, попадающих в пределы окна, называют **размером окна**.

Окно **всегда перемещается в сторону больших номеров**.

Концепция скользящего окна

Идеализированная схема «скольжения» окна передачи.



Реализация метода скользящего окна в TCP

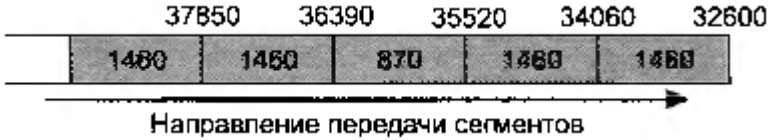
Алгоритм скользящего окна в протоколе TCP имеет существенную особенность – окно определено на множестве **нумерованных байтов** неструктурированного потока данных, передаваемого приложением протоколу TCP.

В ходе переговорного процесса модули TCP обеих участвующих в обмене сторон договариваются между собой о **начальном номере байта** (у каждой стороны он свой), с которого будет вестись отсчет в течение всего функционирования данного соединения.

Нумерация байтов в пределах сегмента осуществляется начиная от заголовка.

Реализация метода скользящего окна в TCP

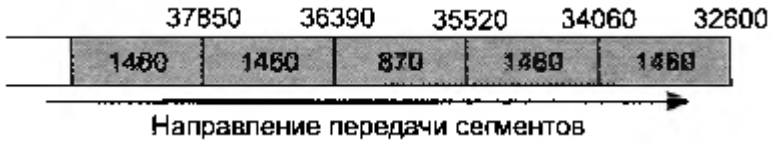
Когда отправитель посылает TCP-сегмент, он помещает в поле **последовательного номера** номер первого байта данного сегмента, который служит **идентификатором** сегмента.



На основании этих номеров получатель TCP-сегмента не только отличает данный сегмент от других, но и позиционирует полученный фрагмент относительно общего потока байтов (он может сделать вывод, например, что полученный сегмент является дубликатом или что между двумя полученными сегментами пропущены данные и т. п.).

Реализация метода скользящего окна в TCP

В качестве **квитанции** получатель сегмента отсылает ответное сообщение (сегмент), в поле **подтвержденного номера** которого он помещает число, **на единицу превышающее максимальный номер байта в полученном сегменте**.



Для первого отправленного сегмента квитанцией о получении (подтвержденным номером) будет число **34060**, для второго – **35520** и т. д.

Подтвержденный номер часто интерпретируют не только как оповещение о благополучной доставке, но и как **номер следующего ожидаемого байта данных**.

Реализация метода скользящего окна в TCP

Квитанция в протоколе TCP посылается только в случае **правильного приема данных** – отсутствие квитанции означает либо потерю сегмента, либо потерю квитанции, либо прием искаженного сегмента.

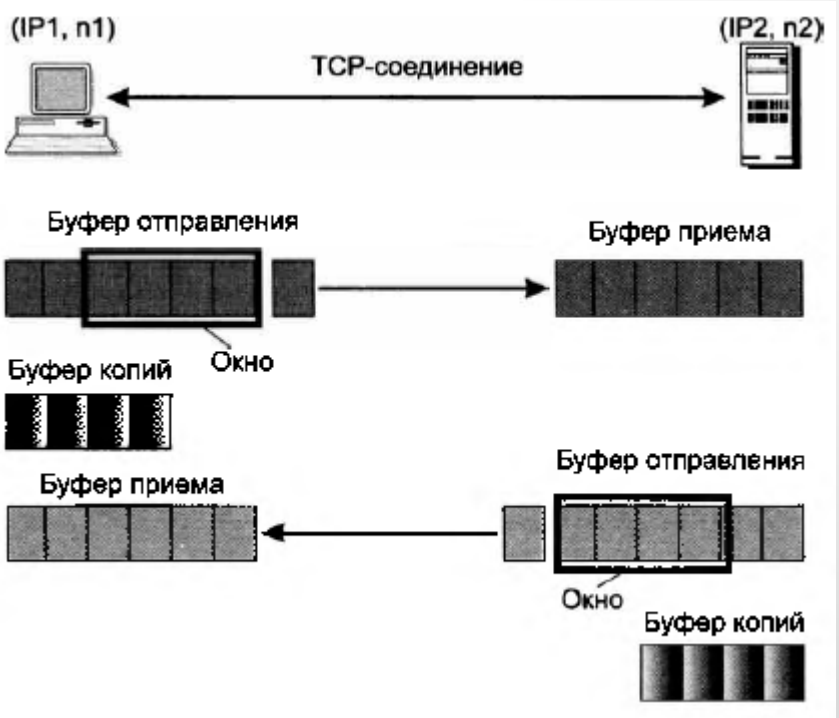
В соответствии с определенным форматом **один и тот же TCP-сегмент** может нести в себе как **пользовательские данные** (в поле данных), так и **квитанцию** (в заголовке), которой подтверждается получение данных от другой стороны.

Реализация метода скользящего окна в TCP

Поскольку протокол TCP является **дуплексным**, каждая сторона одновременно выступает и как **отправитель**, и как **получатель**.

У каждой стороны есть **набор буферов**:

- для **хранения принятых сегментов**;
- для **сегментов, которые только еще предстоит отправить**;
- буфер для **хранения копий сегментов, которые были отправлены, но квитанции о получении которых еще не поступили.**



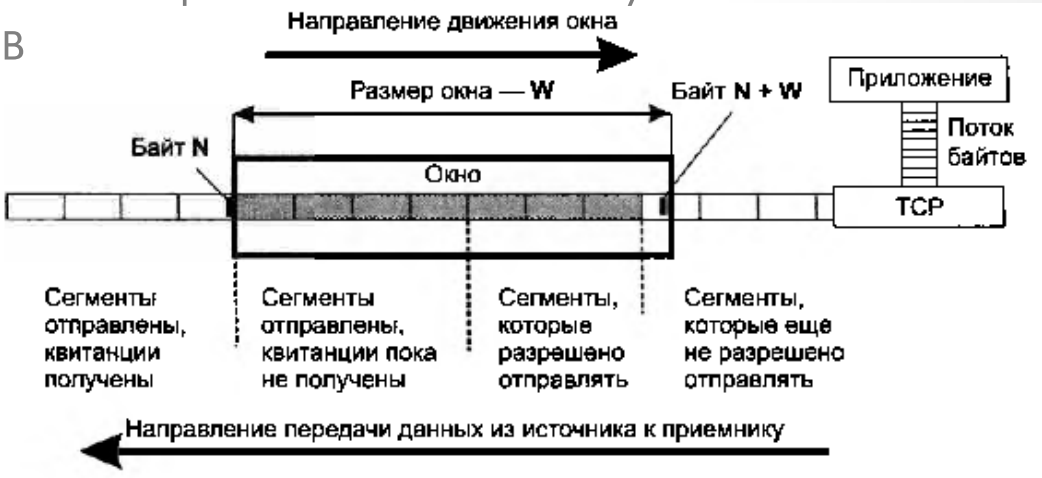
Реализация метода скользящего окна в TCP

1) сегменты, которые уже были отправлены и на которые уже пришли квитанции (последняя квитанция пришла на байт с номером *N*);

2) часть байтов, входящих в окно размером *W* байт, составляют сегменты, которые также уже отправлены, но квитанции на которые пока не получены;

3) сегменты, которые пока не отправлены, но могут быть отправлены, так как входят в пределы окна;

4) сегменты, ни один из которых не может быть отправлен до тех пор, пока не придет очередная квитанция и окно не будет сдвинуто вправо.



Реализация метода скользящего окна в TCP

Накопительный принцип квитирования.

Возможны ситуации, когда сегменты приходят к получателю не в том порядке, в каком были посланы, то есть в приемном буфере может образоваться «**прогалина**».

Получатель может только **еще раз повторить квитанцию** на максимальный байт плюс один в последнем принятом сегменте из непрерывной цепочки сегментов, говоря тем самым, что **все еще ожидает поступления пропущенного блока байтов**.