?????? IPv4 ? IPv6

?????? ?????

Компьютерные сети состоят из множества компонентов и протоколов, работающих совместно. Чтобы понять, как происходит связь между узлами, познакомимся с моделью OSI и моделью TCP/IP. Обе модели помогают визуализировать коммуникацию между узлами.

?????? OSI

Модель открытых систем OSI состоит из 7 уровней и сегодня используется как учебное пособие. Изначально она была задумана как стандартная архитектура для построения сетевых систем, но реальные сети гораздо менее строго структурированы.

- **Уровень 7 (Прикладной)** — протокол, определяющий связь между сервером и клиентом, например, HTTP. Если браузер хочет загрузить изображение, протокол организует и выполняет запрос;
- **Уровень 6 (Представления)** — обеспечивает получение данных в удобном формате. Здесь может происходить шифрование (например, IPSec);
- **Уровень 5 (Сеансовый)** — отвечает за установку, управление и завершение сеансов между клиентом и сервером;
- **Уровень 4 (Транспортный)** — отвечает за сборку и разборку потока данных, делит поток на сегменты с порядковыми номерами и инкапсулирует в протокольный заголовок (TCP, UDP и др.);
- **Уровень 3 (Сетевой)** — отвечает за логическую адресацию устройств, данные инкапсулируются в IP-заголовок и называются "пакетами";
- **Уровень 2 (Канальный)** — данные инкапсулируются в собственный заголовок, например Ethernet 802.3 или беспроводной 802.11, называемый "фреймом", отвечает за управление потоком данных;
- **Уровень 1 (Физический)** — средства передачи данных в виде битов, электрические сигналы и аппаратные интерфейсы;

?????? TCP/IP

Модель TCP/IP выполняет ту же функцию, что и OSI, но лучше подходит для современных сетевых задач. В отличие от 7 уровней OSI, она содержит 4 уровня:

- **Прикладной (4)** — объединяет прикладной, представления и сеансовый уровни OSI, упрощая диагностику;
- **Транспортный (3)** — аналогичен транспортному уровню OSI (протоколы TCP, UDP);
- **Интернет (2)** — аналог сетевого уровня OSI (включая протоколы ARP, IP);

- **Канальный (1)** — также называется уровнем сетевого доступа, включает физический и канальный уровни OSI, отвечает за физическую передачу данных между узлами;

TCP/IP	Модель OSI	Протоколы
Прикладной уровень	Прикладной уровень	DNS, DHCP, HTTP, SSH и др.
—	Уровень представления	JPEG, MPEG, PICT и др.
—	Сеансовый уровень	PAP, SCP, ZIP и др.
Транспортный уровень	Транспортный уровень	TCP, UDP
Интернет уровень	Сетевой уровень	ICMP, IGMP, IPv4, IPv6, IPSec
Канальный уровень	Канальный уровень	ARP, CDP, MPLS, PPP и др.
Физический уровень	Физический уровень	Bluetooth, Ethernet, Wi-Fi и др.

Ethernet

Самым распространенным протоколом канального уровня (уровень 2 модели OSI) в компьютерных сетях является протокол Ethernet. Каждый узел в сети имеет уникальный адрес, называемый MAC-адресом (Media Access Control), иногда его называют "Ethernet-адресом".

MAC-адрес состоит из 48 бит и обычно задан производителем (изменить нельзя), но в последнее время широко используется настройка пользовательских MAC-адресов. RouterOS позволяет задать кастомный MAC-адрес.

Чаще всего MAC-адрес записывается в виде 6 шестнадцатеричных чисел, разделённых двоеточиями, например `D4:CA:6D:01:22:96`.

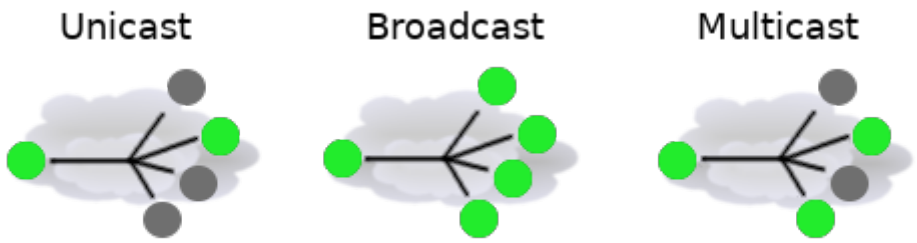
В RouterOS MAC-адрес отображается в конфигурации для всех шиноподобных интерфейсов (Ethernet, Wireless, 60G, VPLS и пр.).

```
[admin@rack1_b32_CCR1036] /interface ethernet> print
Flags: X - disabled, R - running, S - slave
# NAME    MTU MAC-ADDRESS          ARP SWITCH
0 R ether1 1500 D4:CA:6D:01:22:96 enabled
1 R ether2 1500 D4:CA:6D:01:22:97 enabled
2 R ether3 1500 D4:CA:6D:01:22:98 enabled
3 ether4 1500 D4:CA:6D:01:22:99 enabled
4 ether5 1500 D4:CA:6D:01:22:9A enabled
5 ether6 1500 D4:CA:6D:01:22:9B enabled
6 ether7 1500 D4:CA:6D:01:22:9C enabled
7 R ether8 1500 D4:CA:6D:01:22:9D enabled
```

```
8 sfp-sfpplus1 1500 D4:CA:6D:01:22:94 enabled
9 sfp-sfpplus2 1500 D4:CA:6D:01:22:95 enabled
```

???? MAC-???????

- **Unicast** — адрес, отправляется всем узлам в области коллизии (например, кабелю между двумя узлами или всем приёмникам в беспроводной сети). Только узел с совпадающим MAC принимает фрейм (если не включен режим promiscuous).
- **Broadcast** — адрес `FF:FF:FF:FF:FF:FF`, принят и перенаправлен всеми узлами в сети второго уровня.
- **Multicast** — адрес, принимаемый всеми узлами, настроенными на приём сообщений с данным адресом.



IP-??????? ???????????

Протокол Ethernet подходит для передачи данных между двумя узлами в сети Ethernet, но не используется самостоятельно. Для доступа третьего уровня (сетевого) применяется протокол IP для уникальной адресации хостов.

В большинстве современных сетей используются IPv4-адреса, 32-битные, записанные в десятичном формате с точками, например, `192.168.88.1`.

Сетей может быть несколько логических, чтобы определить принадлежность IP адреса к сети, используется маска сети (netmask). Маска задается числом бит, определяющих подсеть, либо десятичной записью, например, 24-битовая маска — `255.255.255.0`.

Рассмотрим адрес `192.168.3.24/24`:

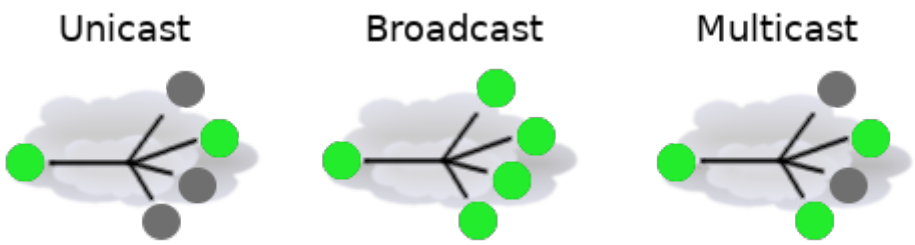
```
11000000 10101000 00000011 00011000  => 192.168.3.24
11111111 11111111 11111111 00000000  => /24 или 255.255.255.0
```

Как видно, старшие 24 бита маскированы, оставляя диапазон адресов от 0 до 255.

Адрес с первым значением в диапазоне используется для идентификации сети (здесь 192.168.3.0), а последний — для широковещательной передачи по сети (здесь 192.168.3.255). Это оставляет диапазон 1..254 для адресации хостов — unicast-адреса.

?????????? IPv4-??????

- **Broadcast** — адрес для рассылки данных сразу всем в сети, например `255.255.255.255` для локального широковещания, или направленное широковещание к адресу сети;
- **Multicast** — адреса из диапазона `224.0.0.0` до `239.255.255.255` для групповых сообщений. Отправитель посылает один пакет на адрес группы, роутеры копируют его для всех подписавшихся участников;



В логической IP-сети unicast, broadcast и multicast визуализируются по-другому.

Существуют зарезервированные адреса, например для частных сетей, которые используются только в локальной сети и обычно не маршрутизируются в Интернет:

- 10.0.0.0/8 — 10.0.0.0 до 10.255.255.255
- 172.16.0.0/12 — 172.16.0.0 до 172.31.255.255
- 192.168.0.0/16 — 192.168.0.0 до 192.168.255.255

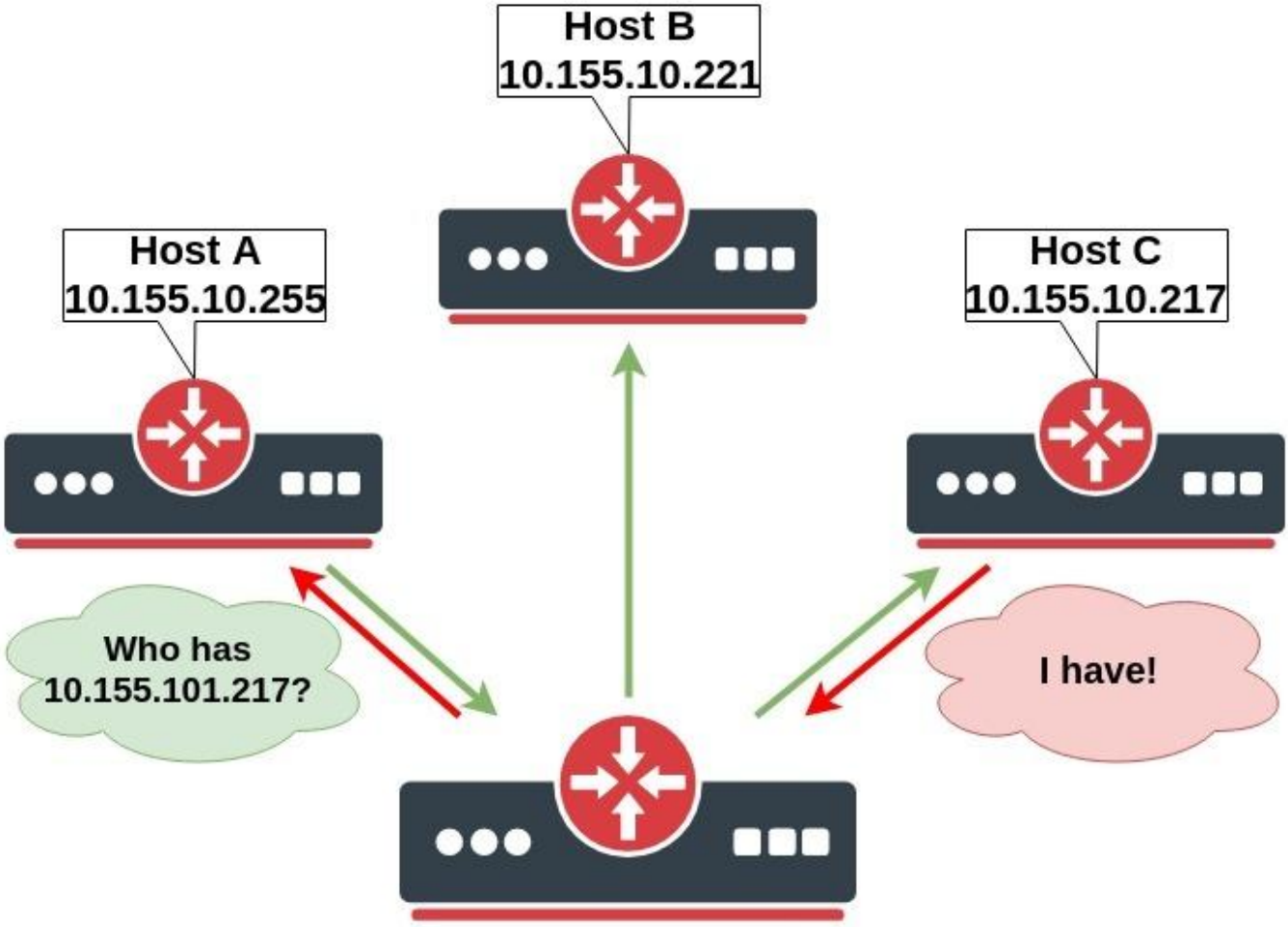
ARP ? ????? ???????

Хотя IP-пакеты адресуются IP-адресами, для физической передачи данных между хостами используются аппаратные адреса (MAC). Для сопоставления IP и MAC применяют протокол ARP (Address Resolution Protocol), описанный в RFC 826.

Каждое устройство хранит таблицу ARP. Обычно она формируется динамически, но может быть частично или полностью статической для повышения безопасности.

IPv6 избавляет от необходимости ARP.

При отправке пакета в локальной сети хост проверяет таблицу ARP для поиска MAC получателя. Если MAC отсутствует, посылается широковещательный ARP-запрос, чтобы получить MAC по IP. Хост с данным IP отвечает своим MAC.



?????? ????????? ? ARP

Добавим IP-адреса на хосты A, B и C:

```
/ip address add address=10.155.101.225 interface=ether1  (Host A)
/ip address add address=10.155.101.221 interface=ether1  (Host B)
/ip address add address=10.155.101.217 interface=ether1  (Host C)
```

Запустим сниффер пакетов и пинг с Host A к Host C:

```
/tool sniffer set file-name=arp.pcap filter-interface=ether1 start
/ping 10.155.101.217 count=1
/stop
```

Скачайте arp.pcap файл и откройте в Wireshark для анализа:

PcsCompu_85:69:b5	Broadcast	ARP	42 Who has 10.155.101.217? Tell 10.155.101.225
PcsCompu_3c:79:3a	PcsCompu_85:69:b5	ARP	42 10.155.101.217 is at 08:00:27:3c:79:3a
10.155.101.225	10.155.101.217	ICMP	70 Echo (ping) request id=0x1c01, seq=0/0, ttl=255 (reply in 428)
10.155.101.217	10.155.101.225	ICMP	70 Echo (ping) reply id=0x1c01, seq=0/0, ttl=64 (request in 427)

- Host A посылает ARP-запрос, кто владеет 10.155.101.217]

- Host C отвечает своим MAC-адресом]
- Оба хоста обновляют ARP таблицы, теперь пинг успешно проходит]

В RouterOS ARP таблицу можно просмотреть командой `/ip arp print`]

```
[admin@host_a] /ip arp> print
Flags: X - disabled, I - invalid, H - DHCP, D - dynamic, P - published, C - complete
# ADDRESS         MAC-ADDRESS          INTERFACE
0 DC 10.155.101.217 08:00:27:3C:79:3A ether1
```

?????? ARP

По умолчанию ARP включён на интерфейсах (Enabled), динамические записи добавляются автоматически.

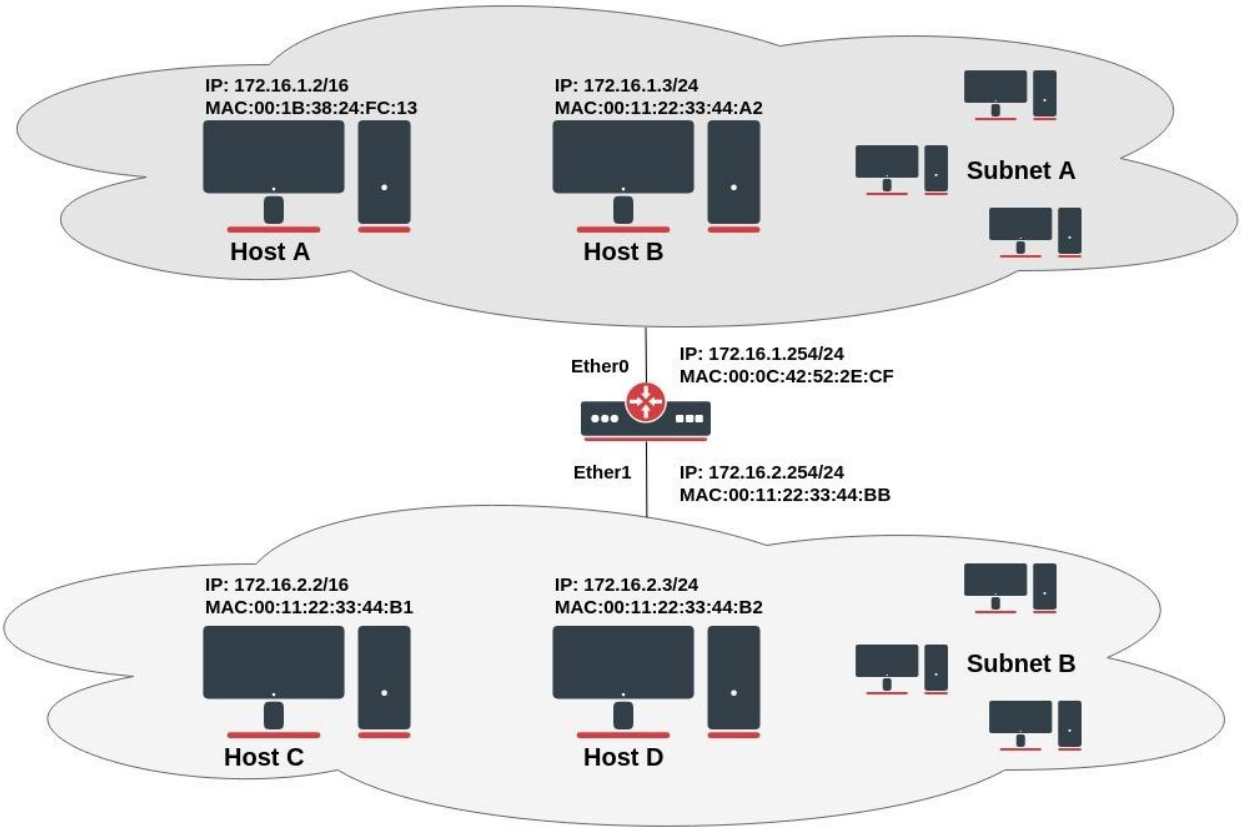
Если ARP отключён (arp=disabled), роутер не отвечает на ARP-запросы клиентов, требуется настройка статической ARP записи на клиентах. Например:

```
/ip arp add mac-address=08:00:27:3C:79:3A address=10.155.101.217 interface=ether1
```

Режим reply-only означает, что роутер отвечает только на ARP-запросы. MAC соседей нужно задавать статически (через /ip arp), но не требуется добавлять MAC адрес роутера в таблицы других хостов как при отключённом ARP.

Proxy ARP

Корректно настроенный proxy ARP позволяет роутеру работать прозрачным ARP прокси между напрямую подключёнными сетями. Это позволяет, например, назначать клиентам dial-in (PPP, PPPoE, PPTP) IP адреса из той же подсети, что и LAN.



Рассмотрим пример настройки на изображении выше. Хост A (172.16.1.2) в подсети A хочет отправить пакеты хосту D (172.16.2.3) в подсети B.
У хоста A маска подсети /16, что означает, что он считает себя напрямую подключённым ко всей сети 172.16.0.0/16 (то есть к одной и той же локальной сети).
Поскольку хост A считает, что адрес назначения находится в той же сети, он отправляет ARP-запрос, чтобы определить MAC-адрес хоста D.
(Если бы хост A понял, что IP-адрес назначения не принадлежит его подсети, он бы отправил пакет на шлюз по умолчанию.)
Хост A рассылает ARP-запрос в подсети A.

Информация из анализатора пакетов:

```
No.     Time       Source             Destination       Protocol  Info
12   5.133205  00:1b:38:24:fc:13  ff:ff:ff:ff:ff:ff  ARP      Who has 173.16.2.3?  Tell
173.16.1.2
```

Детали пакета:

```
Ethernet II, Src: (00:1b:38:24:fc:13), Dst: (ff:ff:ff:ff:ff:ff)
    Destination: Broadcast (ff:ff:ff:ff:ff:ff)
```

```
    Source: (00:1b:38:24:fc:13)
    Type: ARP (0x0806)
Address Resolution Protocol (request)
    Hardware type: Ethernet (0x0001)
    Protocol type: IP (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (0x0001)
    [Is gratuitous: False]
    Sender MAC address: 00:1b:38:24:fc:13
    Sender IP address: 173.16.1.2
    Target MAC address: 00:00:00:00:00:00
    Target IP address: 173.16.2.3
```

С этим ARP-запросом хост A (172.16.1.2) запрашивает у хоста D (172.16.2.3) его MAC-адрес. ARP-запрос инкапсулируется в Ethernet-кадр, где MAC-адрес хоста A используется как адрес источника, а адрес назначения — широковещательный (FF:FF:FF:FF:FF:FF). Широковещательная передача на канальном уровне (Layer 2 broadcast) означает, что кадр будет отправлен всем устройствам в том же домене широковещательной рассылки уровня 2, включая интерфейс ether0 маршрутизатора, но не достигнет хоста D, так как маршрутизатор по умолчанию не пересылает широковещательные кадры уровня 2.

Поскольку маршрутизатор знает, что целевой адрес (172.16.2.3) находится в другой подсети, но может достичь хоста D, он отвечает своему хосту A собственным MAC-адресом.

```
No.     Time       Source             Destination        Protocol   Info
13   5.133378  00:0c:42:52:2e:cf   00:1b:38:24:fc:13    ARP        172.16.2.3 is at
00:0c:42:52:2e:cf
```

Детали пакета:

```
Ethernet II, Src: 00:0c:42:52:2e:cf, Dst: 00:1b:38:24:fc:13
   Destination: 00:1b:38:24:fc:13
   Source: 00:0c:42:52:2e:cf
   Type: ARP (0x0806)
Address Resolution Protocol (reply)
   Hardware type: Ethernet (0x0001)
   Protocol type: IP (0x0800)
   Hardware size: 6
   Protocol size: 4
   Opcode: reply (0x0002)
```

```
   [Is gratuitous: False]
   Sender MAC address: 00:0c:42:52:2e:cf
   Sender IP address: 172.16.1.254
   Target MAC address: 00:1b:38:24:fc:13
   Target IP address: 172.16.1.2
```

Это ответ Proxy ARP, который маршрутизатор отправляет хосту A. Маршрутизатор посылает обратно unicast-ответ ARP со своим MAC-адресом в качестве источника и MAC-адресом хоста A в качестве назначения, как бы говоря: «Отправляй пакеты мне, а я доставлю их туда, куда нужно».

Когда хост A получает ARP-ответ, он обновляет свою ARP-таблицу, как показано ниже:

```
C:\Users\And>arp -a
Interface: 173.16.2.1 --- 0x8
  Internet Address      Physical Address      Type
  173.16.1.254          00-0c-42-52-2e-cf    dynamic
  173.16.2.3            00-0c-42-52-2e-cf    dynamic
  173.16.2.2            00-0c-42-52-2e-cf    dynamic
```

После обновления ARP-таблицы хост A пересылает все пакеты, предназначенные для хоста D (172.16.2.3), напрямую на интерфейс маршрутизатора ether0 (00:0c:42:52:2e:cf), а маршрутизатор передаёт их хосту D. Кэш ARP на хостах в подсети A содержит MAC-адрес маршрутизатора для всех хостов подсети B, поэтому все пакеты, адресованные подсети B, отправляются маршрутизатору, который пересылает их соответствующим узлам в подсети B.

При использовании Proxy ARP несколько IP-адресов разных хостов могут быть сопоставлены с одним MAC-адресом — MAC-адресом маршрутизатора.

Proxy ARP можно включить на каждом интерфейсе отдельно с помощью команды:

```
[admin@MikroTik] /interface ethernet> set 1 arp=proxy-arp
[admin@MikroTik] /interface ethernet> print
Flags: X - disabled, R - running
  #    NAME                 MTU   MAC-ADDRESS         ARP
  0  R ether1               1500  00:30:4F:0B:7B:C1 enabled
  1  R ether2               1500  00:30:4F:06:62:12 proxy-arp
[admin@MikroTik] interface ethernet>
```

Local Proxy ARP
Если свойство `arp` на интерфейсе установлено в `local-proxy-arp`, маршрутизатор выполняет Proxy ARP только для трафика, приходящего и уходящего через этот же интерфейс. В обычной локальной сети два хоста обмениваются данными напрямую, без участия маршрутизатора.

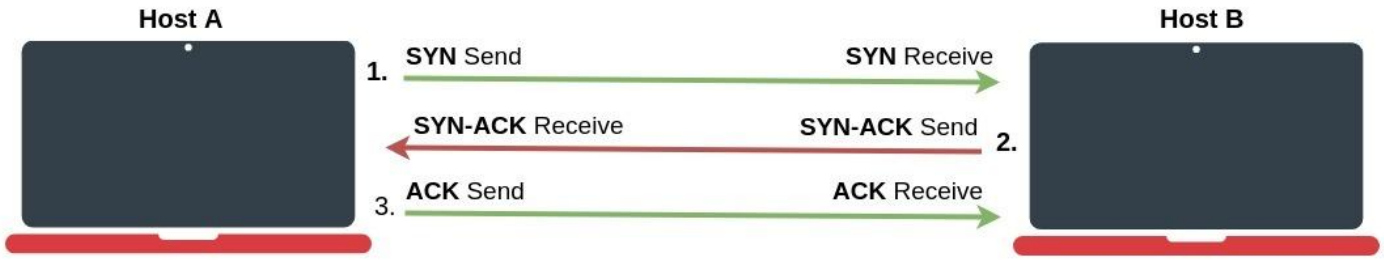
Эта функция используется для поддержки возможностей коммутаторов Ethernet, таких как описанные в RFC 3069, где отдельные порты не могут общаться друг с другом, но могут обмениваться данными с вышестоящим маршрутизатором. Как описано в RFC 3069, можно позволить таким хостам взаимодействовать через маршрутизатор с помощью Proxy ARP. Не обязательно использовать вместе с обычным Proxy ARP.

Эта технология известна под разными названиями:

- В RFC 3069 — **VLAN Aggregation**;
- У Cisco и Allied Telesis — **Private VLAN**;
- У Hewlett-Packard — **Source-Port Filtering** или **Port Isolation**;
- У Ericsson — **MAC-Forced Forwarding** (черновик RFC).

???????????? ? ?????????? TCP ??????

TCP — протокол с установлением соединения, не отправляющий данные до установления соединения. В процессе используется трёхстороннее рукопожатие (three-way handshake), устанавливающее логическое связь с контролем потока и подтверждением доставки.

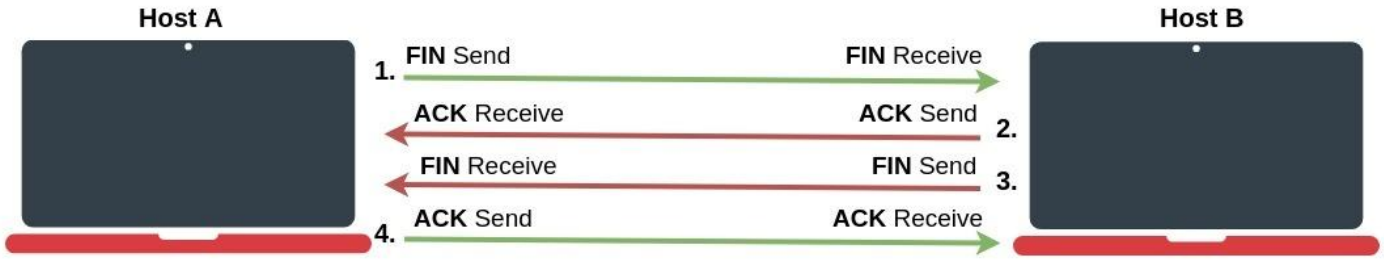


- Host A посылает SYN-пакет с начальным номером последовательности Host B.
- Host B получает SYN и отвечает SYN-ACK.
- Host A получает SYN-ACK и отправляет ACK.
- Host B получает ACK — соединение установлено.

После успешной передачи данных отправитель ждёт подтверждения (ACK). При таймауте пакет пересылается повторно.

?????????? TCP ??????????

Завершение соединения осуществляется четырёхсторонним обменом.



- Host A посылает FIN, сигнализируя о завершении передачи данных.
- Host B входит в состояние CLOSE_WAIT, посылает ACK на FIN. Если у Host B нет данных для передачи, он посылает FIN и переходит в LAST_ACK.
- Host A получает FIN, переходит в TIME_WAIT и посылает ACK.
- Host B получает ACK и соединение завершено.

???????? TCP ????????? (????)

Теперь, когда мы знаем, как устанавливается TCP-соединение, нужно понять, как управляется и поддерживается передача данных. В сетях TCP/IP передача между хостами осуществляется с помощью протокола TCP.

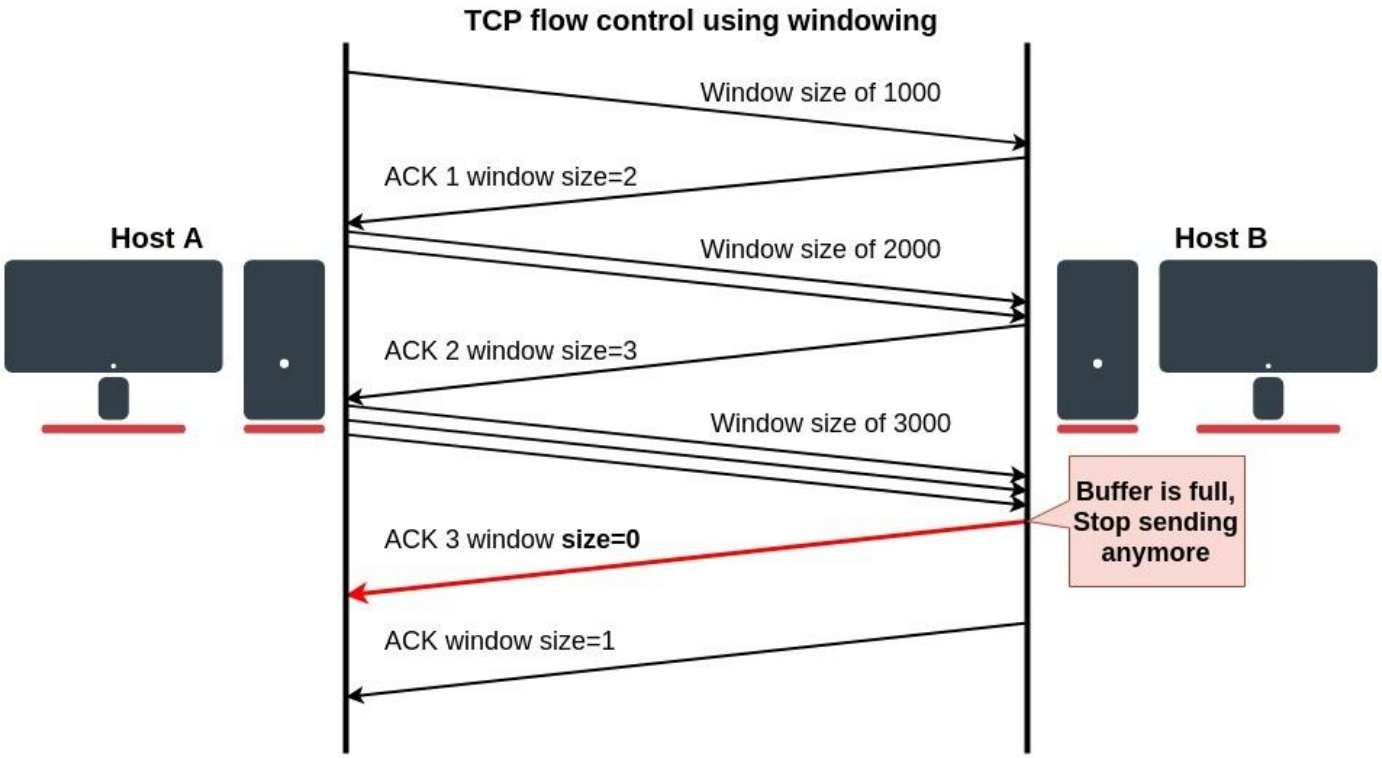
Рассмотрим, что происходит, когда дейтаграммы отправляются быстрее, чем принимающее устройство успевает их обработать. Приёмник сохраняет их в области памяти, называемой буфером. Но поскольку объём буфера ограничен, при его переполнении приёмник начинает отбрасывать кадры. Все отброшенные кадры должны быть переданы повторно, что приводит к снижению производительности передачи данных.

Для решения этой проблемы TCP использует механизм управления потоком (flow control). Для этого применяется окно передачи данных (window mechanism), с помощью которого регулируется объём передаваемой информации.

Когда соединение установлено, приёмник указывает значение поля **window** в каждом TCP-сегменте. Размер окна определяет количество данных, которые приёмник готов временно сохранить в своём буфере. Размер окна (в байтах) передаётся вместе с подтверждениями (ACK) отправителю. Таким образом, размер окна определяет, сколько данных может быть передано от одного хоста к другому без получения подтверждения. Отправитель передаёт только объём данных, соответствующий размеру окна, а затем ждёт подтверждения с обновлённым значением окна.

Если принимающее приложение способно обрабатывать данные так же быстро, как они поступают от отправителя, то приёмник будет отправлять положительные уведомления о размере окна (увеличивая его значение) с каждым подтверждением. Так происходит до тех пор, пока скорость отправителя не превысит скорость обработки приёмника — тогда входящие данные начнут заполнять буфер приёмника, и он отправит подтверждение с **нулевым окном** (zero window). Отправитель, получивший уведомление о нулевом окне, должен приостановить передачу данных, пока не получит новое подтверждение с положительным размером окна.

Рассмотрим иллюстрированный процесс работы оконного механизма:



1. Host A посылает фрейм размером 1000 байт (окно=1000).
2. Host B присылает ACK с размером окна 2000.
3. Host A получает ACK и отправляет два фрейма по 1000 байт.
4. Host B увеличивает окно до 3000, Host A отправляет три фрейма и ждёт подтверждения.
5. Буфер Host B переполняется быстрее, чем данные обрабатываются, окно уменьшается до нуля — нужно ждать.
6. Алгоритмы управления перегрузкой TCP (Reno, Vegas, Tahoe и др.) регулируют размер окна.

Revision #5
Created 11 November 2025 13:37:06 by Admin
Updated 11 November 2025 14:13:55 by Admin