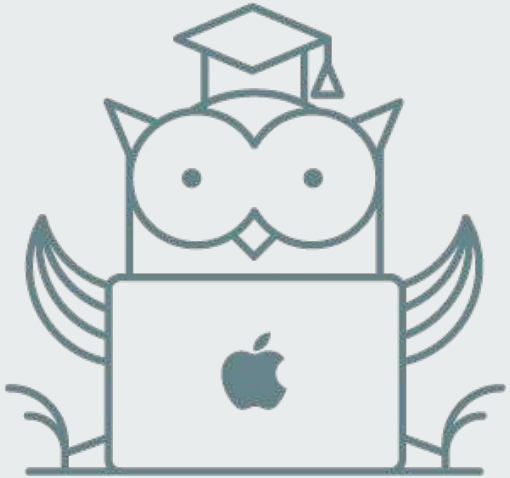
Курс «Администратор Linux»

Сеть: начало

Занятие # 8

Дмитрий Молчанов
Григорий Ожегов

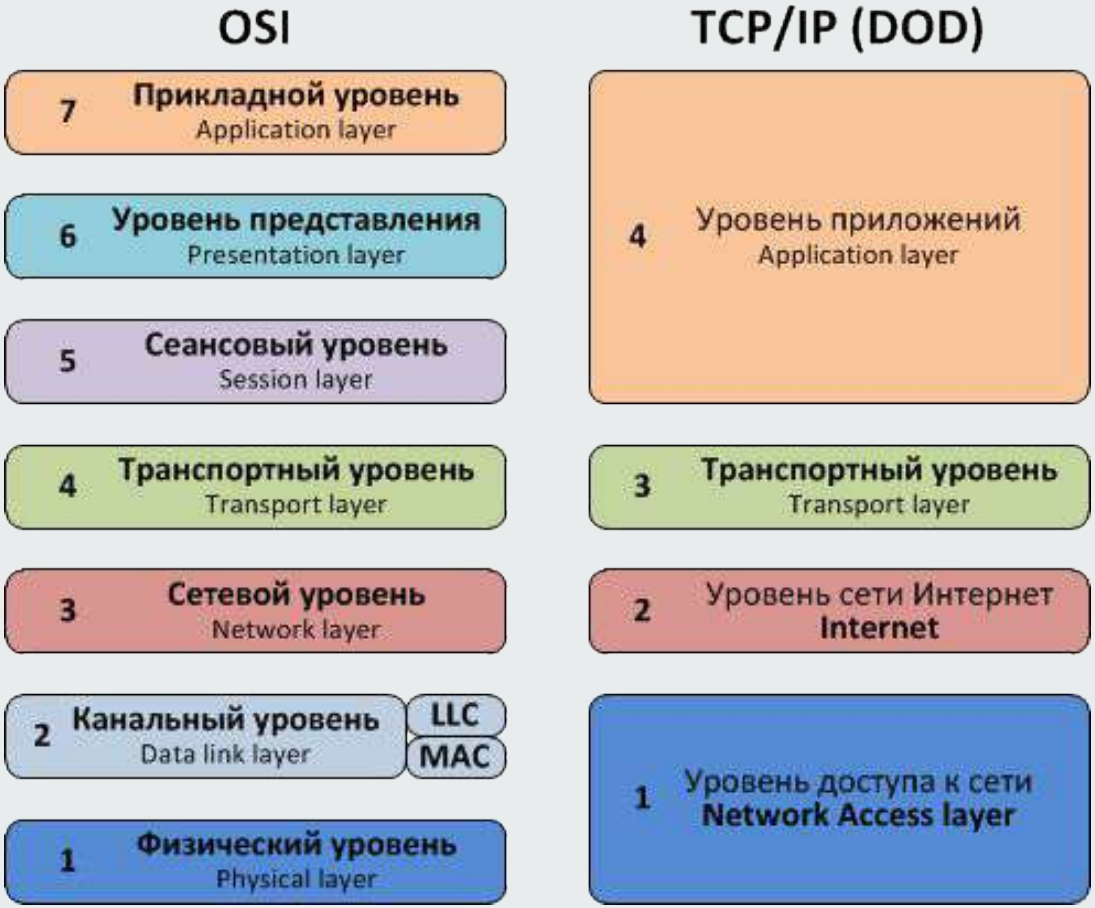


Содержание

- Сетевые уровни
- Сетевые адреса
- Протоколы
- Утилиты для работы с сетью

Немного правды о сетях

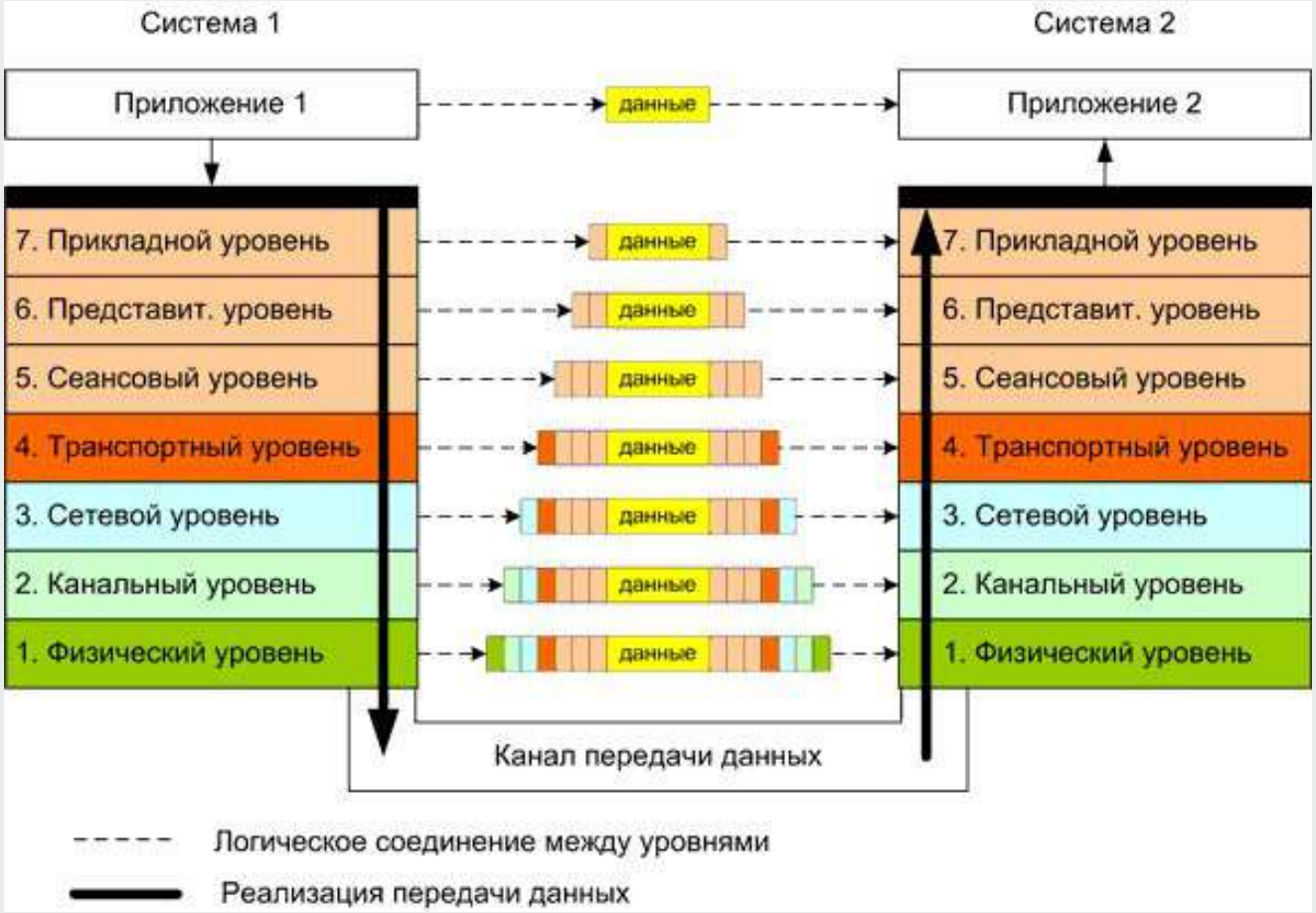
Сеть и слои

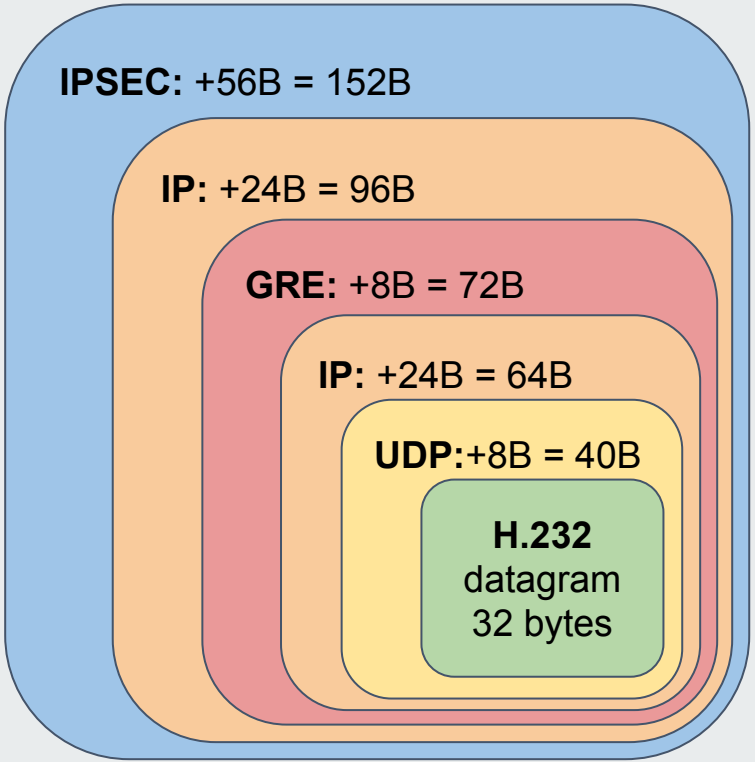

OSI
TCP/IP (DOD)
7 Прикладной уровень
Application layer
6 Уровень представления
Presentation layer
5 Сеансовый уровень
Session layer
4 Транспортный уровень
Transport layer
3 Сетевой уровень
Network layer
2 Канальный уровень
Data link layer
LLC
MAC
1 Физический уровень
Physical layer
4 Уровень приложений
Application layer
3 Транспортный уровень
Transport layer
2 Уровень сети Интернет
Internet
1 Уровень доступа к сети
Network Access layer

Важные моменты.

- Протоколы канального уровня оперируют кадрами и имеют ограниченную длину кадров, что накладывает определенные ограничения на протоколы вышележащих уровней: L3 и выше.
- Каждый уровень добавляет свои накладные расходы, потому у протоколов этих уровней могут быть свои ограничения - например MSS (Maximum Segment Size) в TCP

Накладные расходы (overhead)





64kbit = 8000B, 8000B/32B = 250PPS, 250PPS*(152B-32B)*8 = 240Kbit. x3.75

Адреса в сети IP.

- L1(L1/IP) - порты на оборудовании
- L2(L1/IP) - MAC-адреса (Media Access Control). 6 октетов.
  - 1-3 - тип адреса и тип передачи (2 бита) идентификатор вендора 24 бита.
    - 1:b0 - тип передачи: 0 - unicast; 1 - multicast (многоадресная передача включая broadcast)
    - 1:b1 - тип адреса: локальный или глобальный. Локальный - назначенный администратором.
  - 4-6 - уникальный идентификатор устройства.
- L3(L2/IP) - IP-адрес. 4 байта.
- L4(L3/IP) - IP:PORT. IP - 4 байта. Порт - 2 байта.
- L5-7(L4/IP) - **URL** `protocol://[user:pass]@host:port/uri`

IP-адреса

Так как ip-адрес является 32-битным идентификатором адресное пространство IPv4 имеет емкость 4294967296 адреса. Адреса 0 (32 бита нулей или 0.0.0.0) и 4294967296 (32 бита единиц, или 255.255.255.255) зарезервированы под определение “любого” адреса и широковещательного - соответственно. В любой сети (за исключением вырожденных - p-t-p или одноадресных) первый адрес всегда является адресом сети, а последний - широковещательным.

IP-адреса: маски

В современном мире мы не работаем с одной большой сетью, поэтому адресное пространство поделено на подсети. В этом случае ip-адрес состоит из 2-х частей - адреса сети и адреса хоста. Для того, чтобы отделить их друг от друга используется маска подсети - количество бит в адресе отвечающих за адрес сети.

Указание маски подсети может быть в разных видах:

- /N (например /24) - прямое указание количества бит отвечающих за адрес сети, например /24 - 24 бита, то есть левые 24 бита адреса - это адрес сети, а на идентификатор хоста остается 8 бит, итого в такой сети может быть 256 адресов из них только 254 использовано для хостов.
- 255.255.255.0 - запись аналогичная по смыслу /24.
- 0.0.0.255 - wildcard запись, используется в основном в оборудовании cisco.

IP-адреса: ipcalc

```
# ipcalc 195.239.108.7/26
Address:   195.239.108.7          11000011.11101111.01101100.00 000111
Netmask:   255.255.255.192 = 26   11111111.11111111.11111111.11 000000
Wildcard:  0.0.0.63               00000000.00000000.00000000.00 111111
=>
Network:   195.239.108.0/26       11000011.11101111.01101100.00 000000
HostMin:   195.239.108.1          11000011.11101111.01101100.00 000001
HostMax:   195.239.108.62         11000011.11101111.01101100.00 111110
Broadcast: 195.239.108.63         11000011.11101111.01101100.00 111111
Hosts/Net: 62                      Class C
```

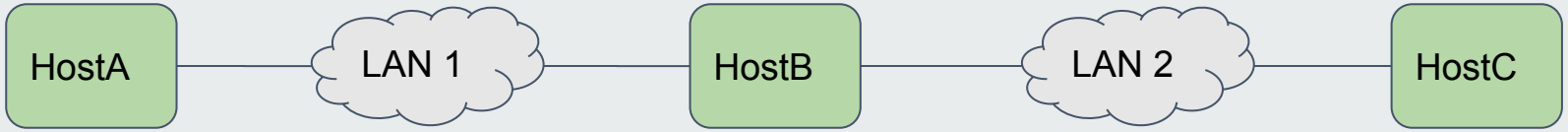
Протоколы

- L2+L3:
  - ARP
- L3:
  - IPv4, IPv6
  - IPX
  - ICMP ?
- L4:
  - TCP
  - UDP
  - GRE
- L5-7:
  - HTTP
  - SMTP
  - DNS
  - NTP
  - ...

ARP

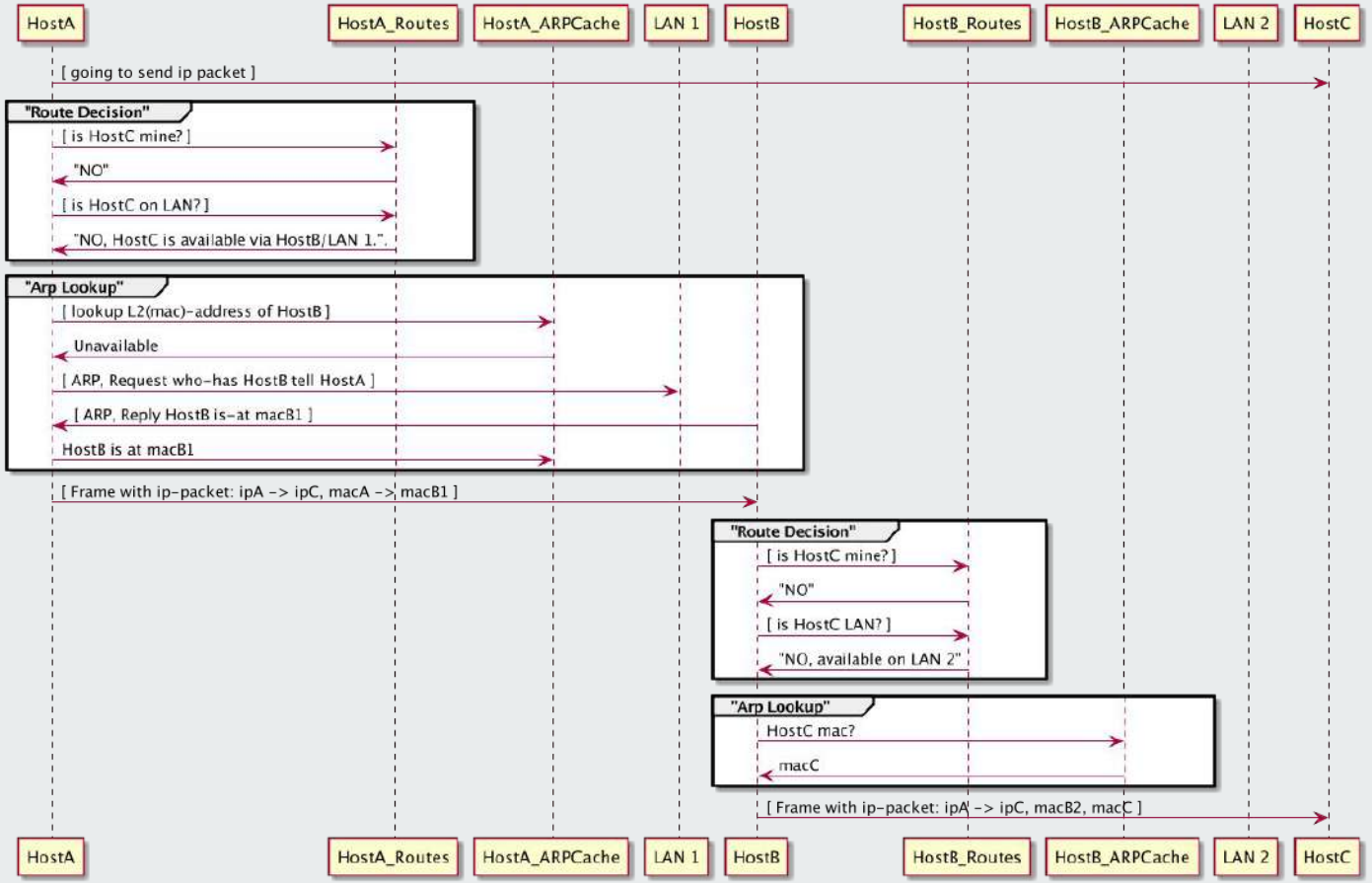
Протокол ARP (Address Resolution Protocol) служит для разрешения адресов (поиска соответствия mac-адресов и IP-адресов) в пределах одного L2-сегмента (L2-домена).

Предположим у нас есть сеть:

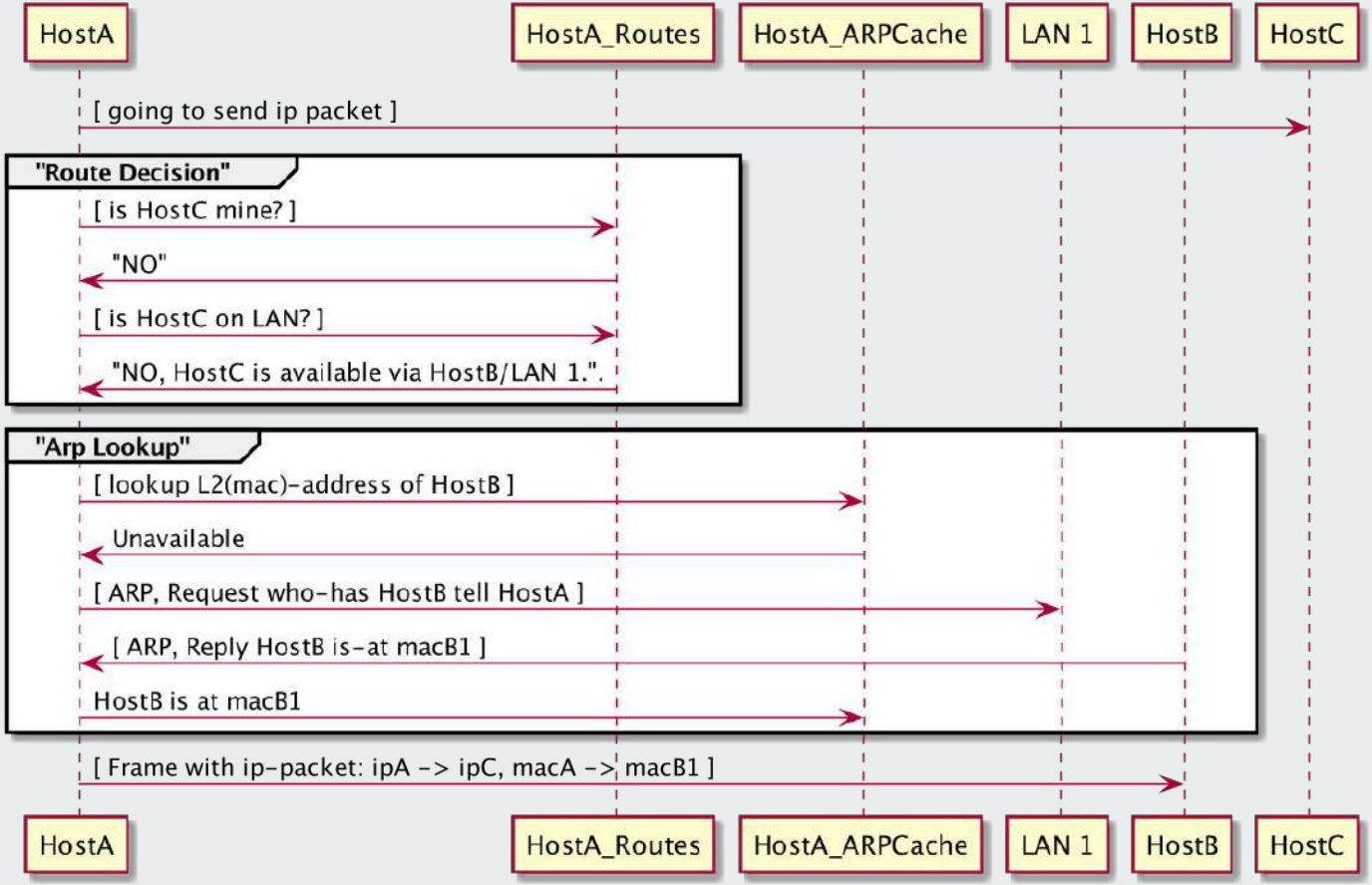


HostA посылает HostC ip-пакет

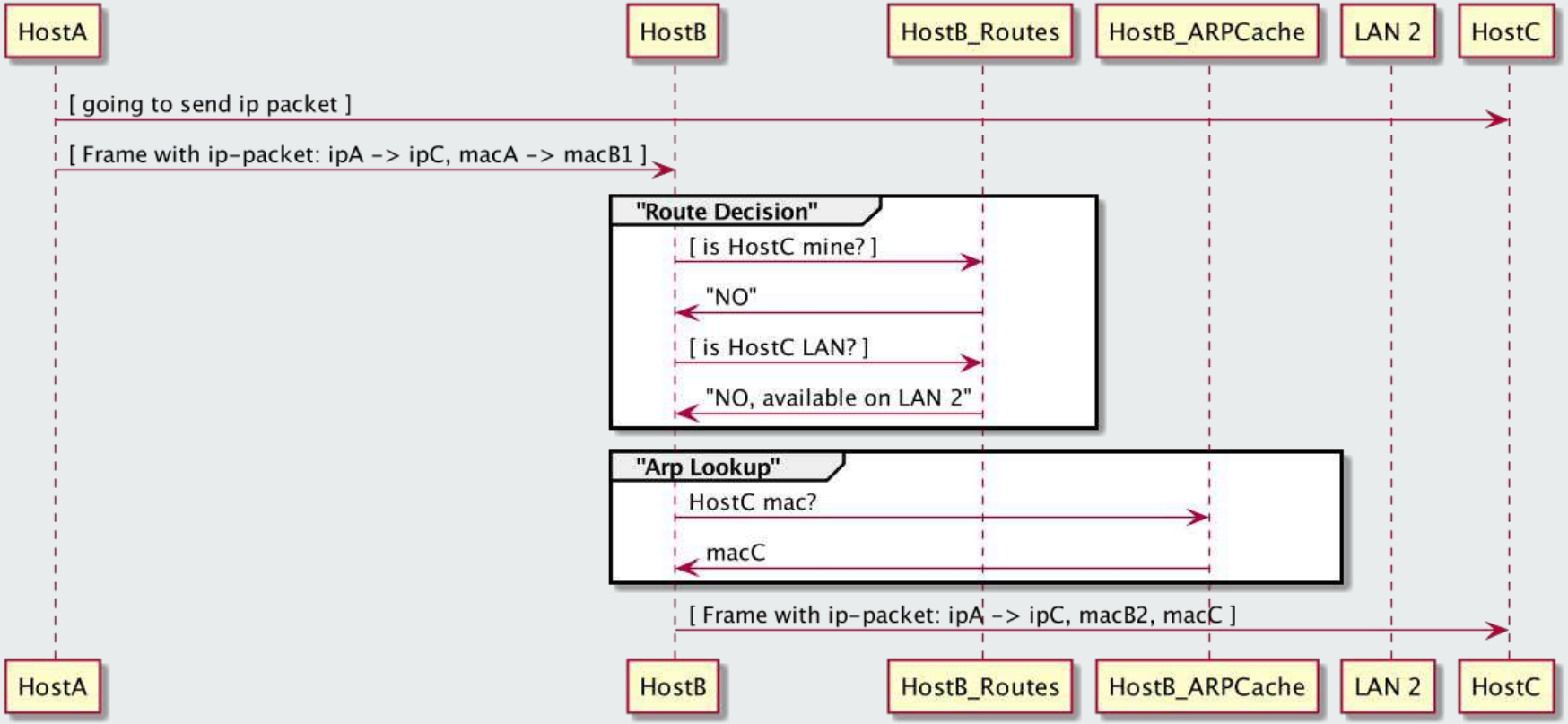
ARP


HostA
HostA_Routes
HostA_ARPCache
LAN 1
HostB
HostB_Routes
HostB_ARPCache
LAN 2
HostC
[going to send ip packet]
"Route Decision"
[is HostC mine?]
"NO"
[is HostC on LAN?]
"NO, HostC is available via HostB/LAN 1.".
"Arp Lookup"
[lookup L2(mac)-address of HostB]
Unavailable
[ARP, Request who-has HostB tell HostA]
[ARP, Reply HostB is-at macB1]
HostB is at macB1
[Frame with ip-packet: ipA -> ipC, macA -> macB1]
"Route Decision"
[is HostC mine?]
"NO"
[is HostC LAN?]
"NO, available on LAN 2"
"Arp Lookup"
HostC mac?
macC
[Frame with ip-packet: ipA -> ipC, macB2, macC]
HostA
HostA_Routes
HostA_ARPCache
LAN 1
HostB
HostB_Routes
HostB_ARPCache
LAN 2
HostC

ARP


HostA
HostA_Routes
HostA_ARPCache
LAN 1
HostB
HostC
[going to send ip packet]
"Route Decision"
[is HostC mine?]
"NO"
[is HostC on LAN?]
"NO, HostC is available via HostB/LAN 1.".
"Arp Lookup"
[lookup L2(mac)-address of HostB]
Unavailable
[ARP, Request who-has HostB tell HostA]
[ARP, Reply HostB is-at macB1]
HostB is at macB1
[Frame with ip-packet: ipA -> ipC, macA -> macB1]
HostA
HostA_Routes
HostA_ARPCache
LAN 1
HostB
HostC

ARP


HostA
HostB
HostB_Routes
HostB_ARPCache
LAN 2
HostC
[going to send ip packet]
[Frame with ip-packet: ipA -> ipC, macA -> macB1]
"Route Decision"
[is HostC mine?]
"NO"
[is HostC LAN?]
"NO, available on LAN 2"
"Arp Lookup"
HostC mac?
macC
[Frame with ip-packet: ipA -> ipC, macB2, macC]
HostA
HostB
HostB_Routes
HostB_ARPCache
LAN 2
HostC

Протокол IP (Internet Protocol) работает поверх канального уровня и, как понятно из названия - InterNet, обеспечивает передачу данных между сетями. В отличие от IPX он является маршрутизируемым протоколом по умолчанию, без дополнительных ухищрений.

Пакеты IP обладают следующими атрибутами хранящимися в заголовке:

- id - идентификатор пакета
- flags - (3 бита): 0:reserve, 1:DF (don’t fragment), 2:has more fragments.
- **ip_src** - Адрес источника
- **ip_dst** - Адрес назначения
- **proto** - инкапсулированый протокол вышележащего уровня (tcp/udp/icmp и т. д.)
- **ttl** - время жизни пакета
- … (еще ряд параметров)

TCP

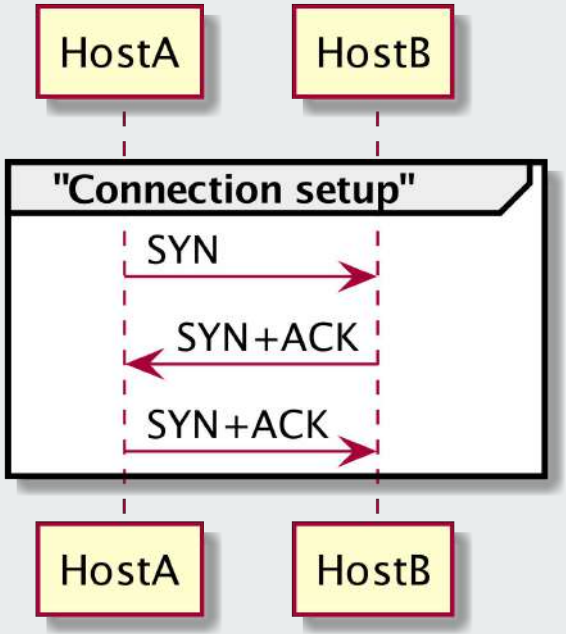
Протокол TCP (Transmission Control Protocol) - протокол с контролем передачи. Предназначен для надежной передачи данных. Это реализуется с помощью механизмов:

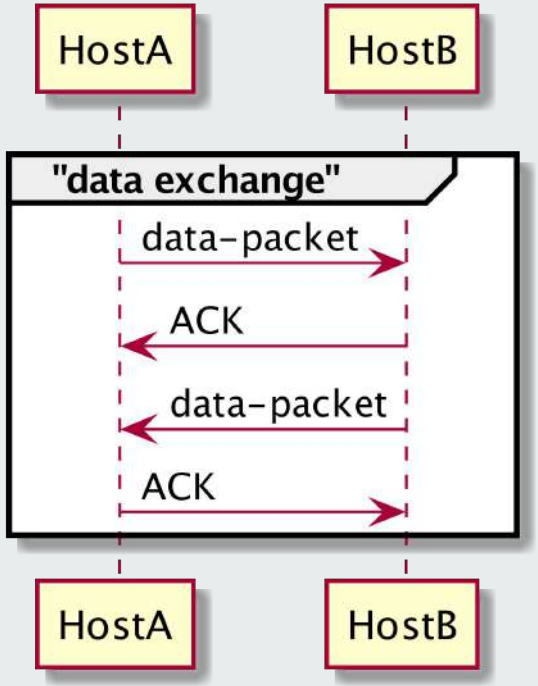
- Установки соединения
- Подтверждения передачи
- Закрытия соединения

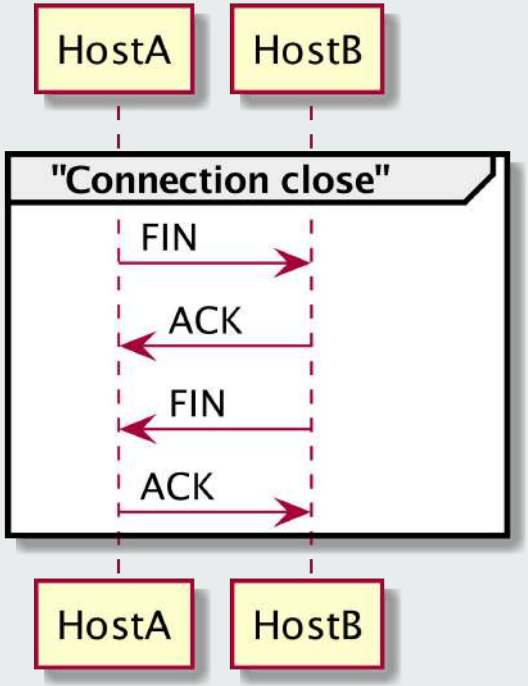
Как результат протокол TCP обеспечивает:

- “контроль доставки”
- Сохранение очередности пакетов

TCP


HostA
HostB
"Connection setup"
SYN
SYN+ACK
SYN+ACK
HostA
HostB


HostA
HostB
"data exchange"
data-packet
ACK
data-packet
ACK
HostA
HostB


HostA
HostB
"Connection close"
FIN
ACK
FIN
ACK
HostA
HostB

UDP

Протокол UDP (User Datagram Protocol) предназначен для простой передачи данных без установки соединения и подтверждения доставки. Поэтому его нельзя считать надежным, но у него есть плюсы:

- На передачу одной датаграммы (влезающей в MTU) ему необходим один пакет.
- Так как доставка датаграммы не гарантируется из-за отсутствия подтверждения передачи - отсутствует время передачи (время между отправкой данных и подтверждением их получения), что позволяет эффективно использовать UDP для потоковых протоколов которые не критичны к потерям или критичны ко времени выполнения операции, например DNS.

ICMP

Одним из наиважнейших протоколов IP является ICMP (Internet Control Message Protocol). Его задача - передача служебных сообщений протокола IP, а также диагностических. Из-за того, что диагностические сообщения могут влиять на маршрутизацию трафика этот протокол часто запрещают, что приводит к неправильным последствиями.

Примеры полезных Служебных Сообщений:

- TTL Expired
- Destination host unreachable
- Fragmentation needed

Пример диагностических сообщений:

- Echo request
- Echo reply

Утилиты ping и traceroute построены на использовании протокола ICMP

Протокол добавляет к ip свои заголовки: icmp_type и icmp_code.

Протоколы L5-7

Большинство базовых протоколов которые мы используем в повседневной жизни - текстовые и диалоговые. К таким протоколам относятся:

- http
- smtp
- pop3
- imap

Пример с HTTP

```
[root@linux-demo ~]# curl -vI http://mail.ru
* About to connect() to mail.ru port 80 (#0)
*   Trying 217.69.139.201...
* Connected to mail.ru (217.69.139.201) port 80 (#0)
> HEAD / HTTP/1.1
> User-Agent: curl/7.29.0
> Host: mail.ru
> Accept: */*
>
< HTTP/1.1 301 Moved Permanently
HTTP/1.1 301 Moved Permanently
< Server: nginx/1.10.3
```

Немного об именах интерфейсов

- Старая нотация eth0, eth1… *iftypeN*. Группировка интерфейсов по типу и сквозная нумерация. Из глобальных минусов - в качестве eth0 может оказаться не тот интерфейс, что до перезагрузки, например, если вставить новую карточку в “младший” слот.
- Новая нотация (от systemd) - Predictable Network Interface Names. В своем виде по умолчанию использует форматы (упрощенно)

```
(en|wl)[P<domain>]p<bus>s<slot>[f<function>][n<phys_port_name>|d<dev_port>] - PCI location
(en|wl)[P<domain>]o<bus>[f<function>][n<phys_port_name>|d<dev_port>] - Onboard device
```

Таким образом enp0s3 говорит нам о том, что мы имеем дело с Ethernet-адаптером подключенным к шине pci №0 в слот №3, а eno1 говорит об onboard ethernet-адаптере с индексом 1.

iproute2

За последние несколько лет пакет iproute плотно интегрировался во все дистрибутивы linux. Он несет с собой несколько утилит:

- ip - управление маршрутизацией, интерфейсами, arp-таблицами
- tc - traffic control - управлением приоритезацией трафика
- ss - sockstat - информация о socket'ах (одна из сторон netstat)
- nstat - информация о сетевых каунтерах

tcpdump/ngrep

Сетевые снифферы.

- tcpdump - информация о сетевой активности. работает максимально близко к “проводу”
- ngrep - утилита для поиска пакетов по содержимому, Network grep. По смыслу схожа с tcpdump.

Спасибо за внимание

Дмитрий Молчанов
Григорий Ожегов