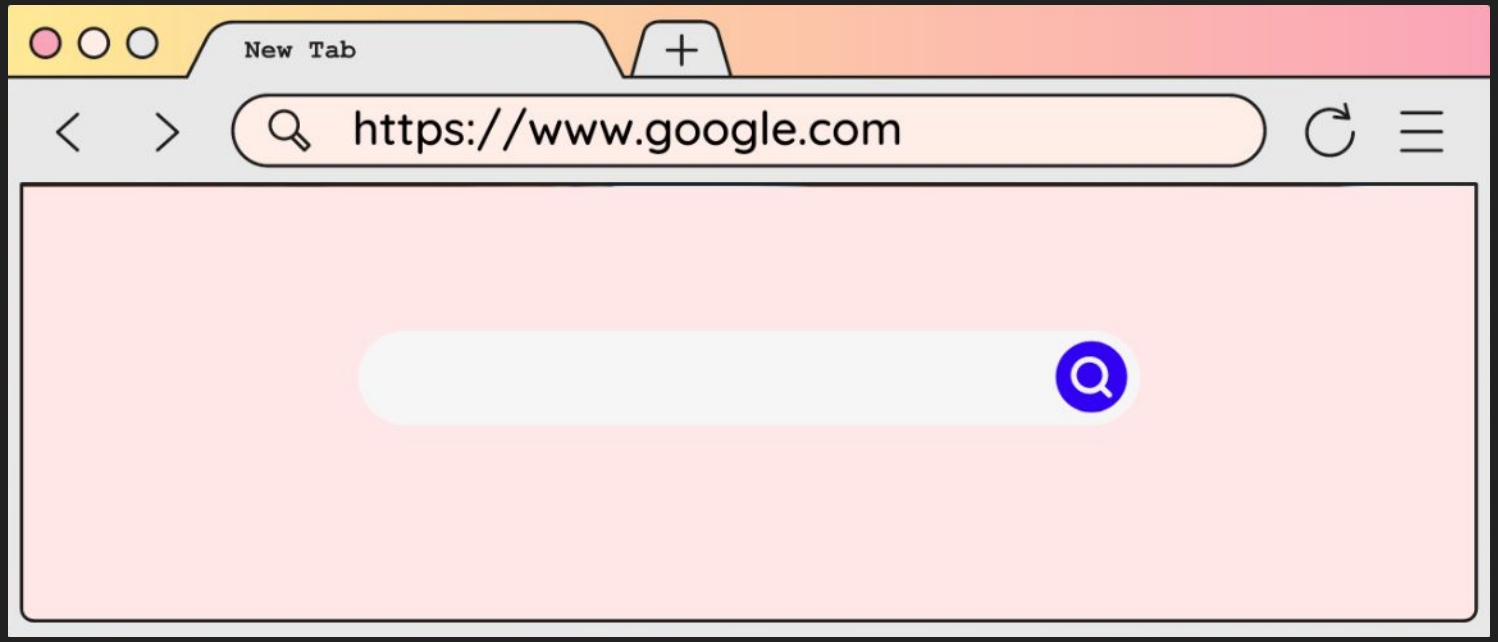
Сети


New Tab
https://www.google.com

HTTP

HTTP запрос

```
Метод URI HTTP/Версия
```

```
GET /cat.jpg HTTP/1.1
```

HTTP запрос с заголовками

```
Метод URI HTTP/Версия

Заголовок: Значение заголовка
```

```
GET /index.html HTTP/1.1

Host: example.com

DNT: 1
```

HTTP запрос с заголовками и телом

```
POST /cat.php HTTP/1.1
Host: example.com
DNT: 1
Content-Length: 13

HELLO, WORLD!
```

HTTP ответ

```
HTTP/Версия Код ответа Пояснение
```

```
HTTP/1.1 200 OK
```

```
HTTP/1.1 404 Not Found
```

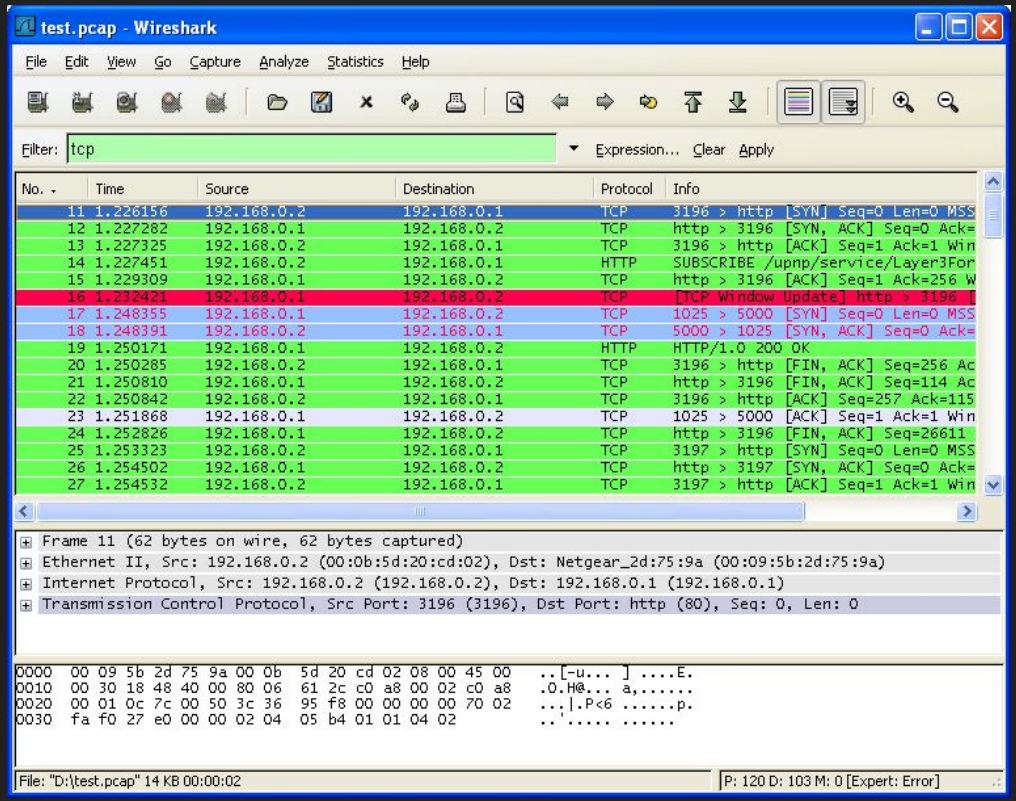
t.me/yet_another_telegram_blog/58

HTTP ответ с заголовками и телом

```
HTTP/1.1 200 OK
Server: nginx/1.2.3
Content-Length: 13

HELLO, WORLD!
```

Wireshark


test.pcap - Wireshark
File Edit View Go Capture Analyze Statistics Help
Filter: tcp
Expression... Clear Apply
No. Time Source Destination Protocol Info
11 1.226156 192.168.0.2 192.168.0.1 TCP 3196 > http [SYN] Seq=0 Len=0 MSS
12 1.227282 192.168.0.1 192.168.0.2 TCP http > 3196 [SYN, ACK] Seq=0 Ack=
13 1.227325 192.168.0.2 192.168.0.1 TCP 3196 > http [ACK] Seq=1 Ack=1 Win
14 1.227451 192.168.0.2 192.168.0.1 HTTP SUBSCRIBE /upnp/service/Layer3For
15 1.229309 192.168.0.1 192.168.0.2 TCP http > 3196 [ACK] Seq=1 Ack=256 W
16 1.232421 192.168.0.1 192.168.0.2 TCP [TCP Window Update] http > 3196 [
17 1.248355 192.168.0.1 192.168.0.2 TCP 1025 > 5000 [SYN] Seq=0 Len=0 MSS
18 1.248391 192.168.0.2 192.168.0.1 TCP 5000 > 1025 [SYN, ACK] Seq=0 Ack=
19 1.250171 192.168.0.1 192.168.0.2 HTTP HTTP/1.0 200 OK
20 1.250285 192.168.0.2 192.168.0.1 TCP 3196 > http [FIN, ACK] Seq=256 Ac
21 1.250810 192.168.0.1 192.168.0.2 TCP http > 3196 [FIN, ACK] Seq=114 Ac
22 1.250842 192.168.0.2 192.168.0.1 TCP 3196 > http [ACK] Seq=257 Ack=115
23 1.251868 192.168.0.1 192.168.0.2 TCP 1025 > 5000 [ACK] Seq=1 Ack=1 Win
24 1.252826 192.168.0.1 192.168.0.2 TCP http > 3196 [FIN, ACK] Seq=26611
25 1.253323 192.168.0.2 192.168.0.1 TCP 3197 > http [SYN] Seq=0 Len=0 MSS
26 1.254502 192.168.0.1 192.168.0.2 TCP http > 3197 [SYN, ACK] Seq=0 Ack=
27 1.254532 192.168.0.2 192.168.0.1 TCP 3197 > http [ACK] Seq=1 Ack=1 Win
Frame 11 (62 bytes on wire, 62 bytes captured)
Ethernet II, Src: 192.168.0.2 (00:0b:5d:20:cd:02), Dst: Netgear_2d:75:9a (00:09:5b:2d:75:9a)
Internet Protocol, Src: 192.168.0.2 (192.168.0.2), Dst: 192.168.0.1 (192.168.0.1)
Transmission Control Protocol, Src Port: 3196 (3196), Dst Port: http (80), Seq: 0, Len: 0
0000 00 09 5b 2d 75 9a 00 0b 5d 20 cd 02 08 00 45 00 ..[-u...] ....E.
0010 00 30 18 48 40 00 80 06 61 2c c0 a8 00 02 c0 a8 .0.H@... a,......
0020 00 01 0c 7c 00 50 3c 36 95 f8 00 00 00 00 70 02 ...|.P<6 ......p.
0030 fa f0 27 e0 00 00 02 04 05 b4 01 01 04 02 ..'..... ......
File: "D:\test.pcap" 14 KB 00:00:02
P: 120 D: 103 M: 0 [Expert: Error]


LEARNED WIRESHARK
HACKERMAN

HTTP/1.1

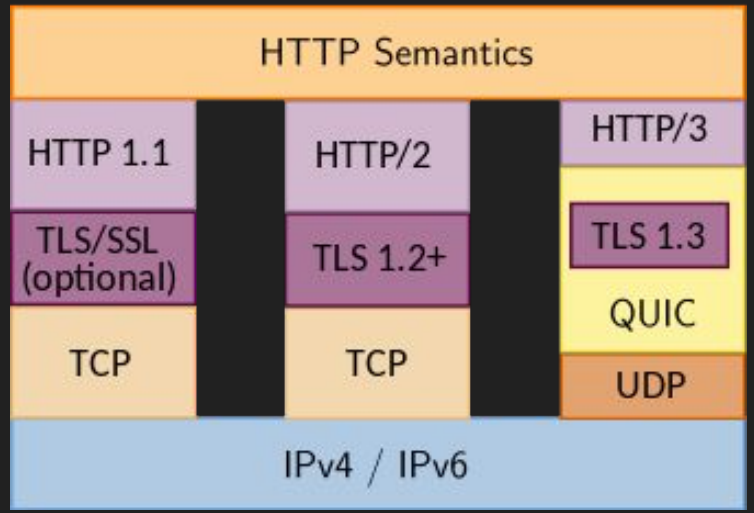
- Keep-Alive
- Cache control

HTTP/2

- Multiplexing
- Binary

HTTP/3

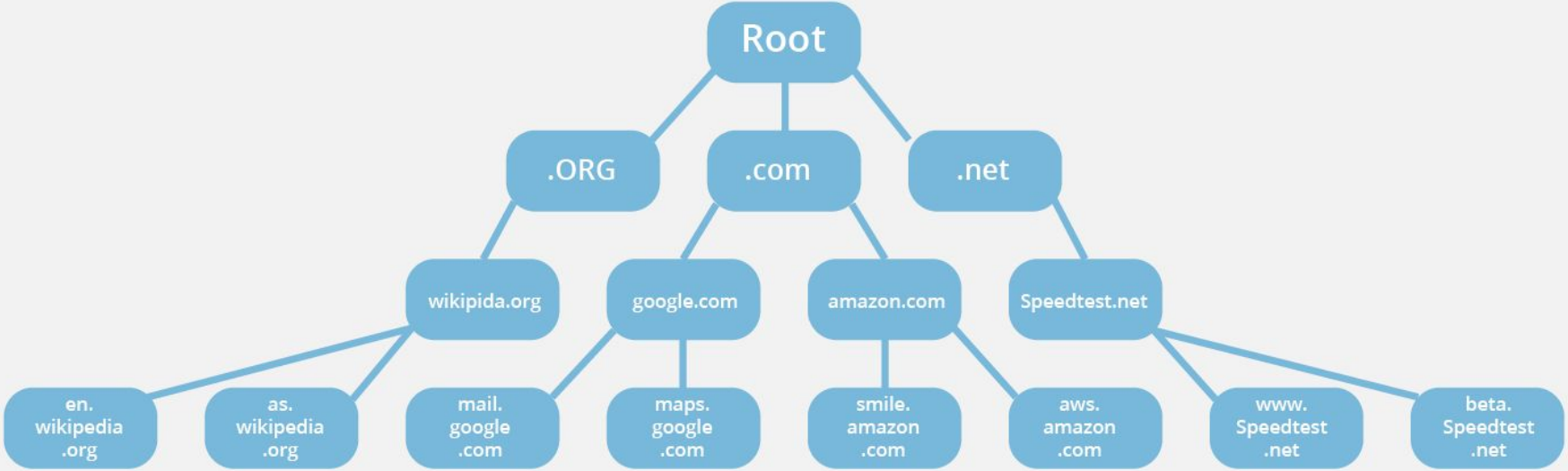
- On QUIC
- 0-rtt



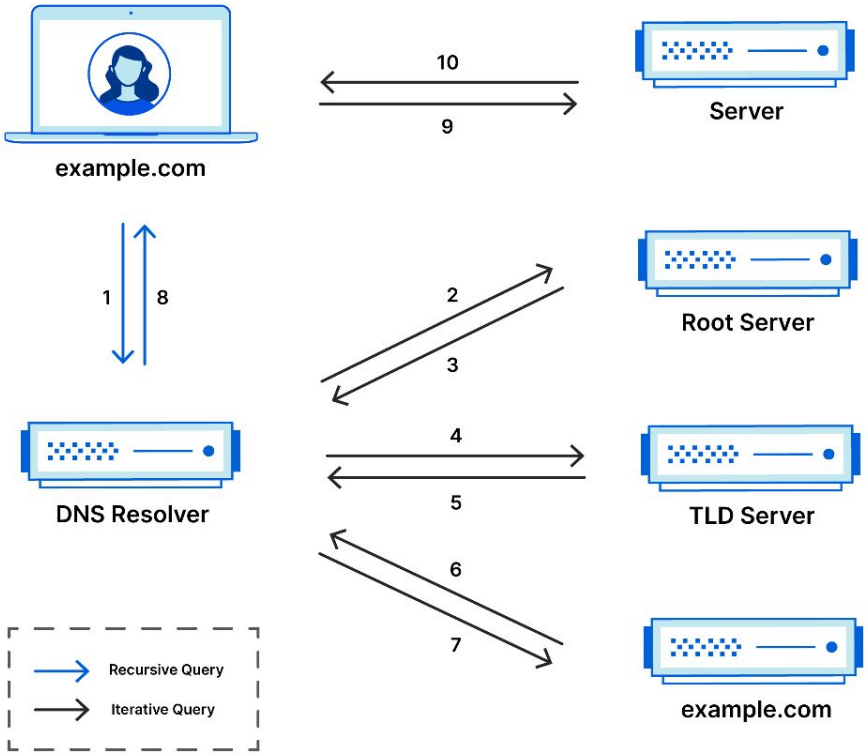

DNS
ЦИФРОВАЯ И
БЫТОВАЯ ТЕХНИКА

sub.example.com.

DNS Зоны


Root
.ORG
.com
.net
wikipida.org
google.com
amazon.com
Speedtest.net
en.
wikipedia
.org
as.
wikipedia
.org
mail.
google
.com
maps.
google
.com
smile.
amazon
.com
aws.
amazon
.com
www.
Speedtest
.net
beta.
Speedtest
.net

Complete DNS Lookup and Webpage Query



Корневые сервера

List of Root Servers

HOSTNAME	IP ADDRESSES	OPERATOR
a.root-servers.net	198.41.0.4, 2001:503:ba3e::2:30	Verisign, Inc.
b.root-servers.net	170.247.170.2, 2801:1b8:10::b	University of Southern California, Information Sciences Institute
c.root-servers.net	192.33.4.12, 2001:500:2::c	Cogent Communications
d.root-servers.net	199.7.91.13, 2001:500:2d::d	University of Maryland
e.root-servers.net	192.203.230.10, 2001:500:a8::e	NASA (Ames Research Center)
f.root-servers.net	192.5.5.241, 2001:500:2f::f	Internet Systems Consortium, Inc.
g.root-servers.net	192.112.36.4, 2001:500:12::d0d	US Department of Defense (NIC)
h.root-servers.net	198.97.190.53, 2001:500:1::53	US Army (Research Lab)
i.root-servers.net	192.36.148.17, 2001:7fe::53	Netnod
j.root-servers.net	192.58.128.30, 2001:503:c27::2:30	Verisign, Inc.
k.root-servers.net	193.0.14.129, 2001:7fd::1	RIPE NCC
l.root-servers.net	199.7.83.42, 2001:500:9f::42	ICANN
m.root-servers.net	202.12.27.33, 2001:dc3::35	WIDE Project

Виды DNS записей

- **A** — задает преобразование имени хоста в IP-адрес.
- **MX** — определяет почтовый ретранслятор для доменного имени, т.е. узел, который обработает или передаст дальше почтовые сообщения, предназначенные адресату в указанном домене.
- **CNAME** — определяет отображение псевдонима в каноническое имя узла.
- **TXT** — содержит общую текстовую информацию. Эти записи могут использоваться в любых целях.
- **AAAA** — Как **A**, только IPV6
- **NS** — определяют DNS-серверы, которые являются авторитативными для данной зоны.

dig

DNS Cache poisoning (spoofing)

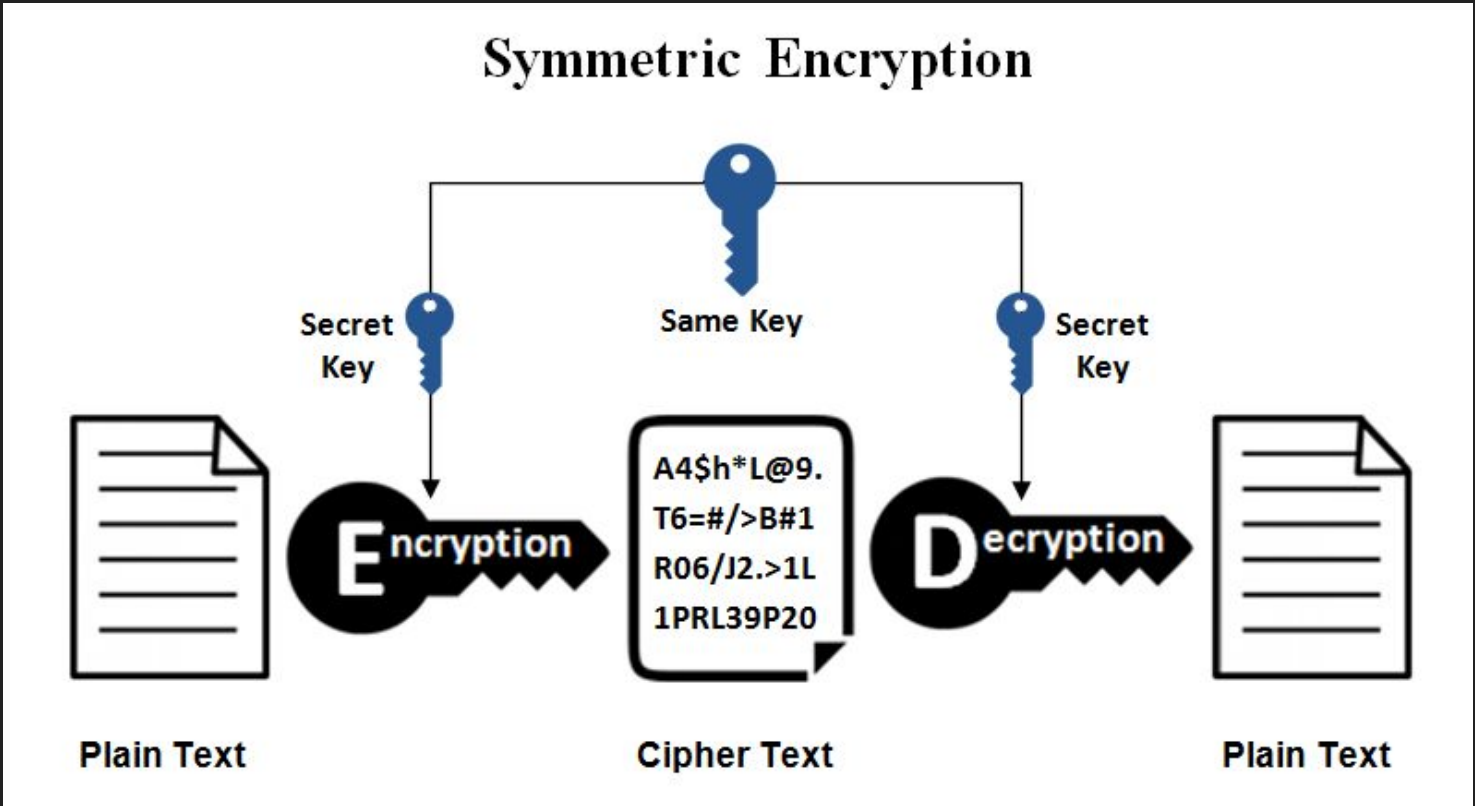
Шифрование DNS

- DNS не шифруется!
- Потенциальные расширения с шифрованием:
  - DNS over HTTPS (DoH, :443)
  - DNS over TLS (DoT, :853)

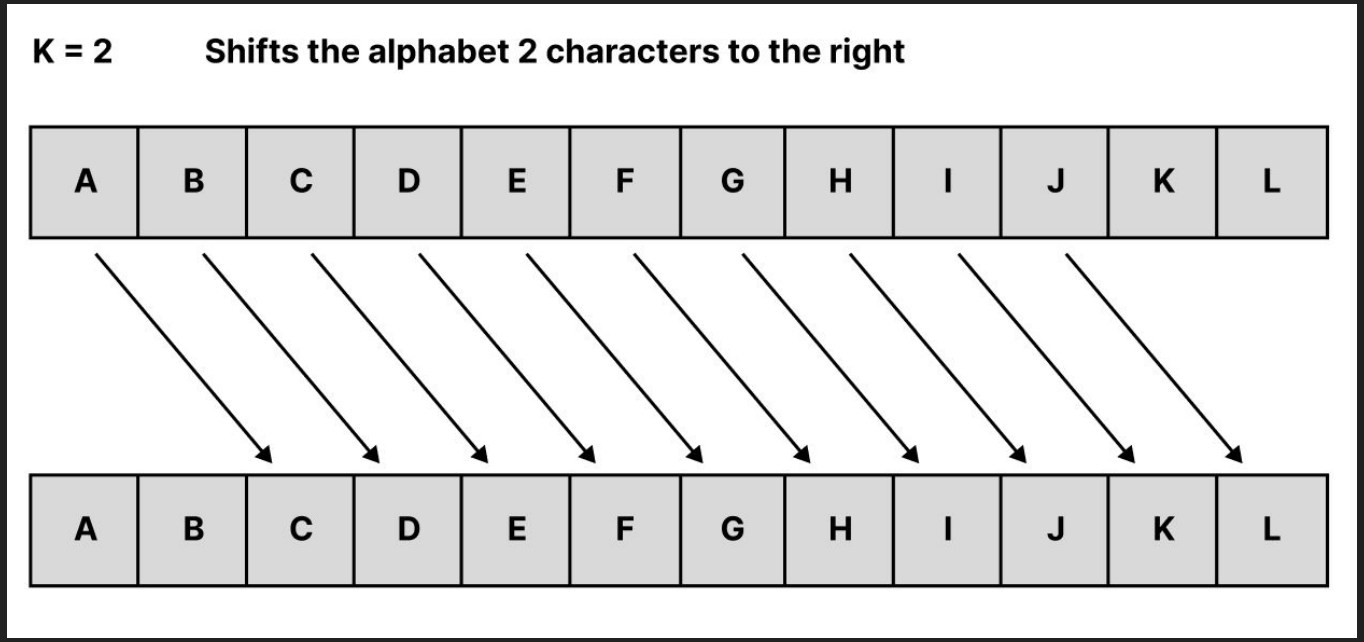
DNSSEC

Асимметричное + симметричное шифрование

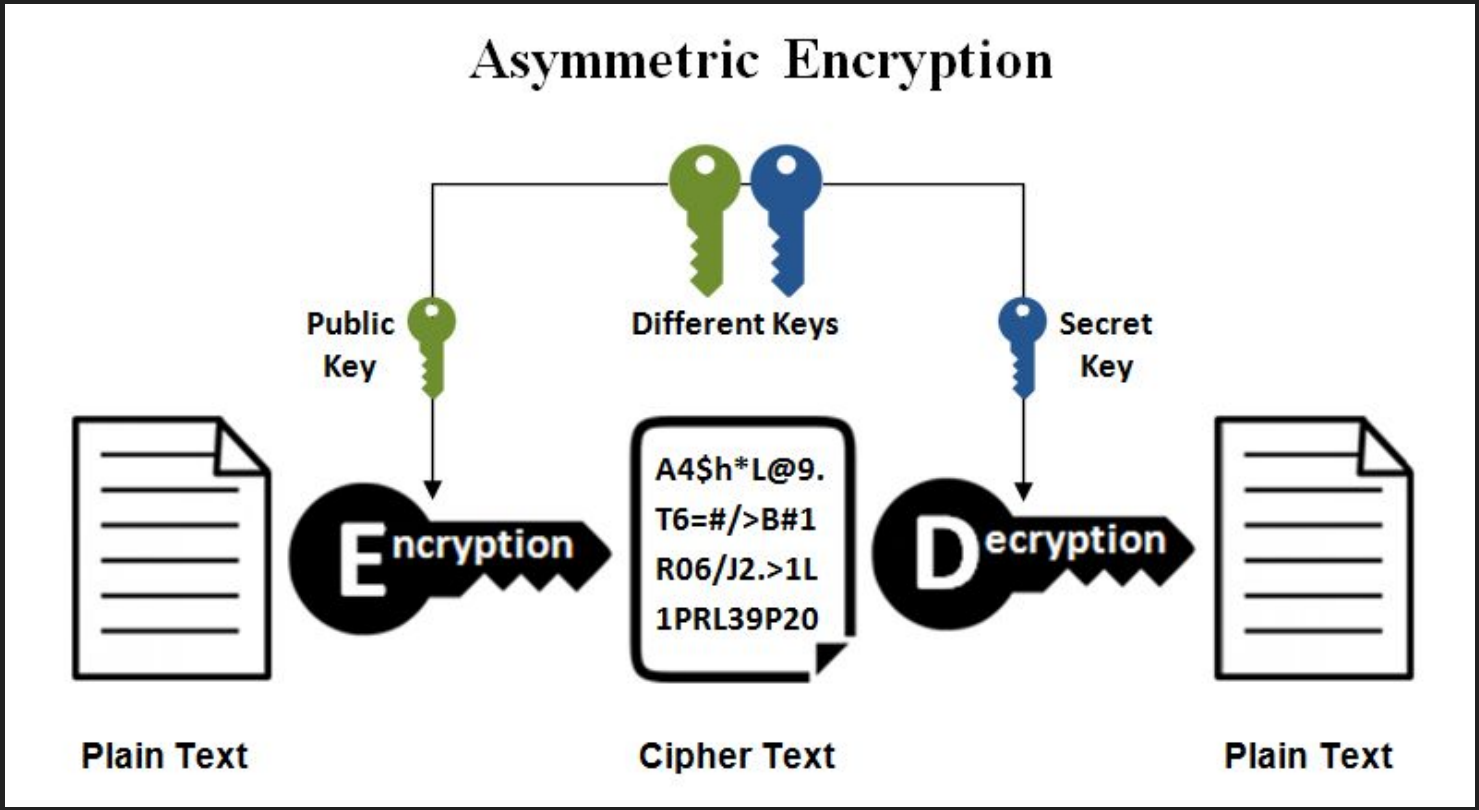
Симметричное шифрование



Симметричное шифрование - Шифр цезаря



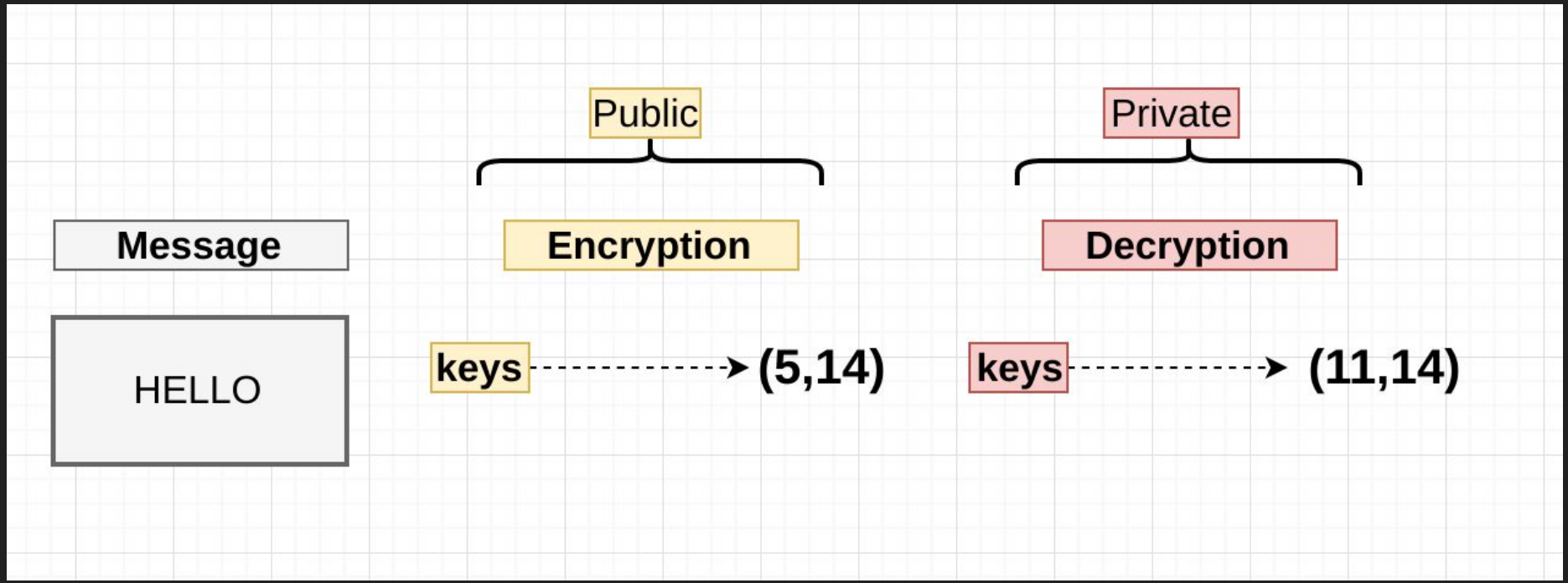
Асимметричное шифрование



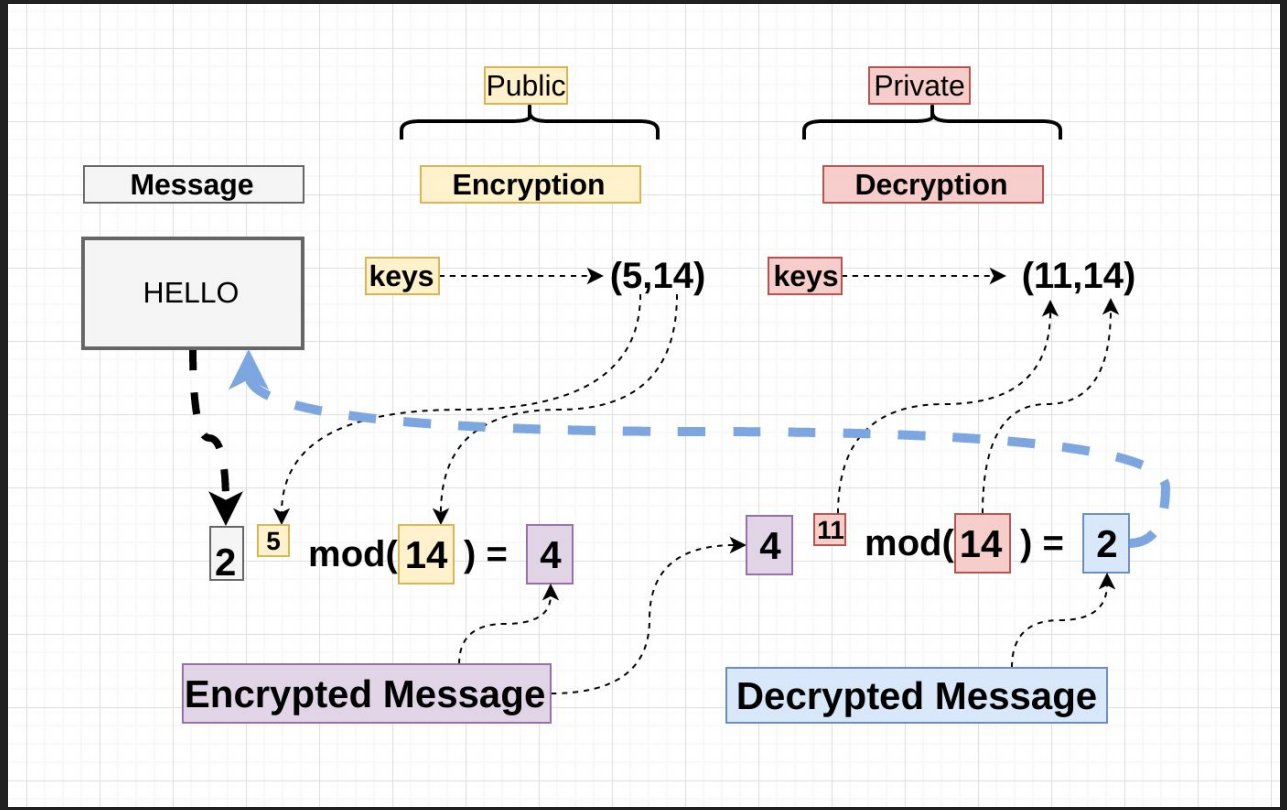
Ассиметричное шифрование - RSA

- Берем *p* и *q* - два простых числа. *N = p * q*
- Считаем *φ(N) = (p - 1) * (q - 1)*
- Берем *e < φ(N) : gcd(e, φ(N)) = 1*
- Считаем *d = e^-1 mod φ(N)*
- Приватный ключ: *(e, N)*
- Публичный ключ: *(d, N)*
- Шифрование: *C = M^e mod N*
- Расшифровка: *M = C^d mod N*

Асимметричное шифрование - RSA

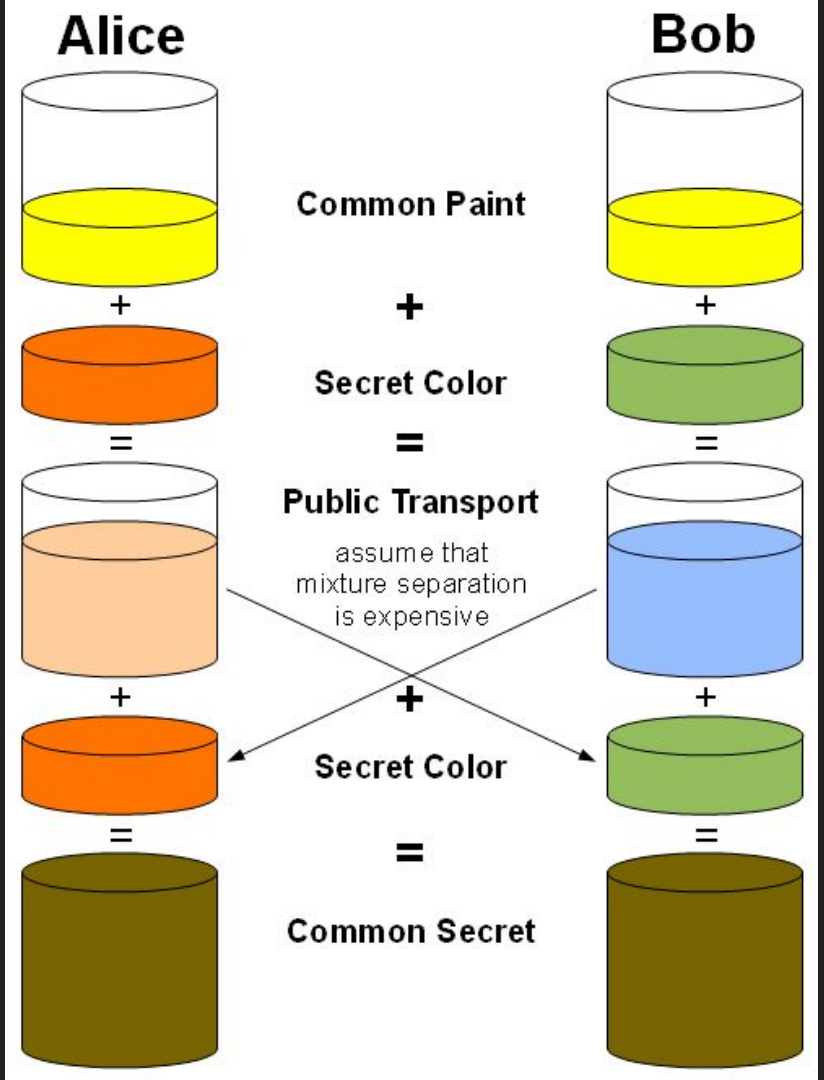


Асимметричное шифрование - RSA



Обмен ключами

Diffie-Hellman
RSA



P – простое, g – перв. корень

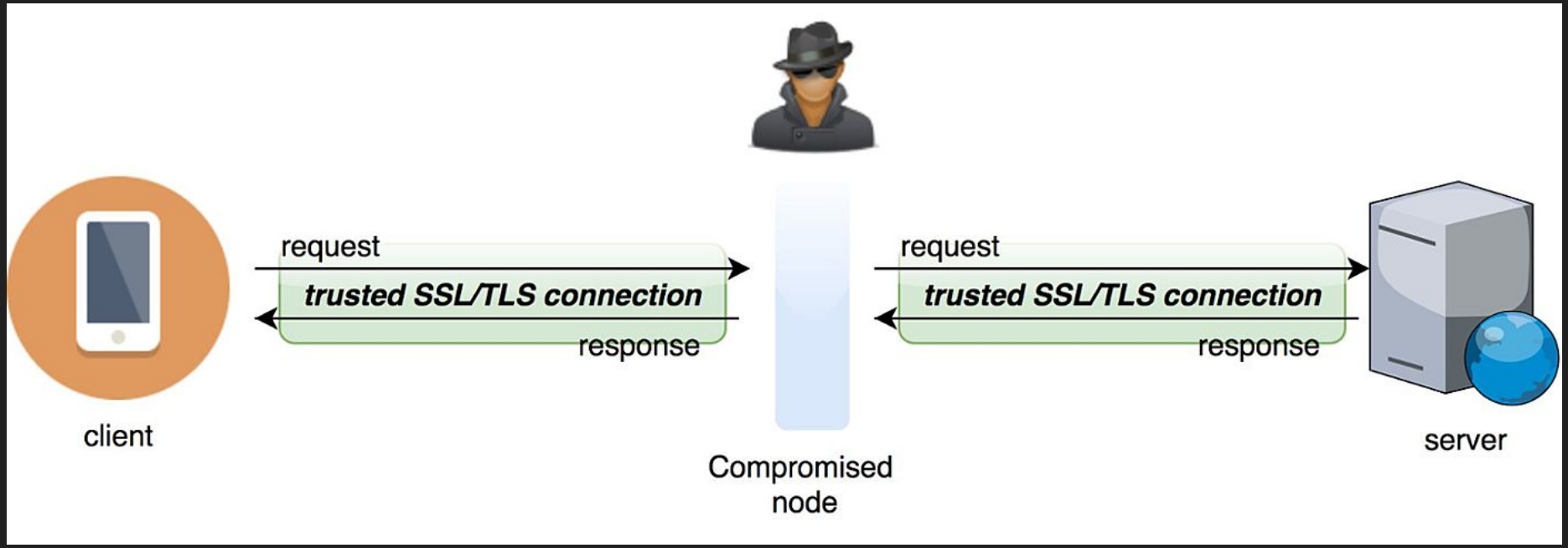
a – случайные числа – b

A = g^a, *B = g^b*

K = B^a *K = A^b*

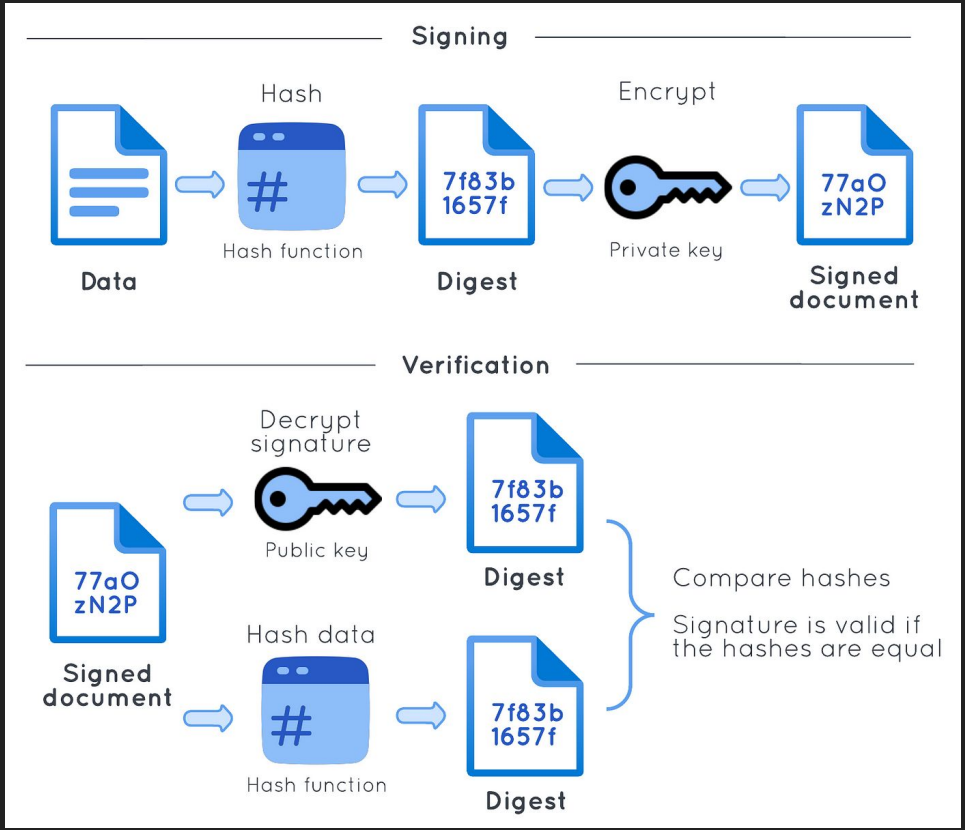
B^a = g^ab = A^b

Проблема аутентификации

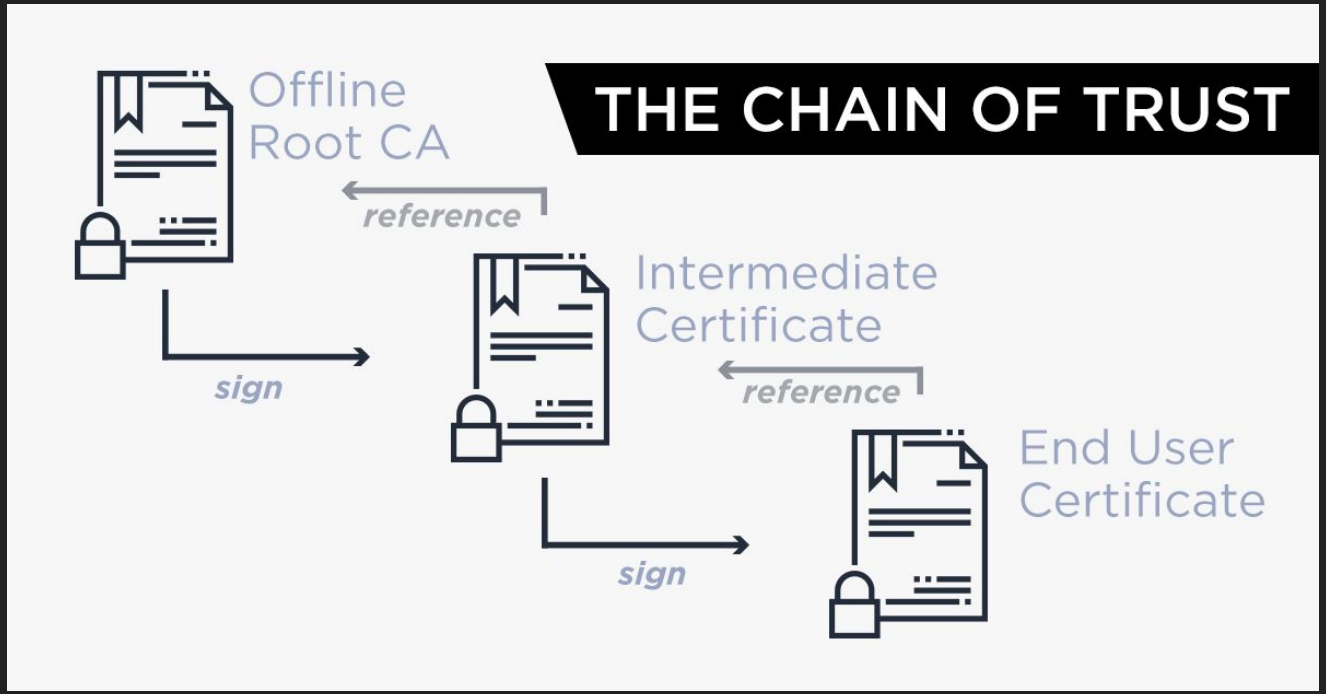


Аутентификация и сертификаты

Цифровая подпись



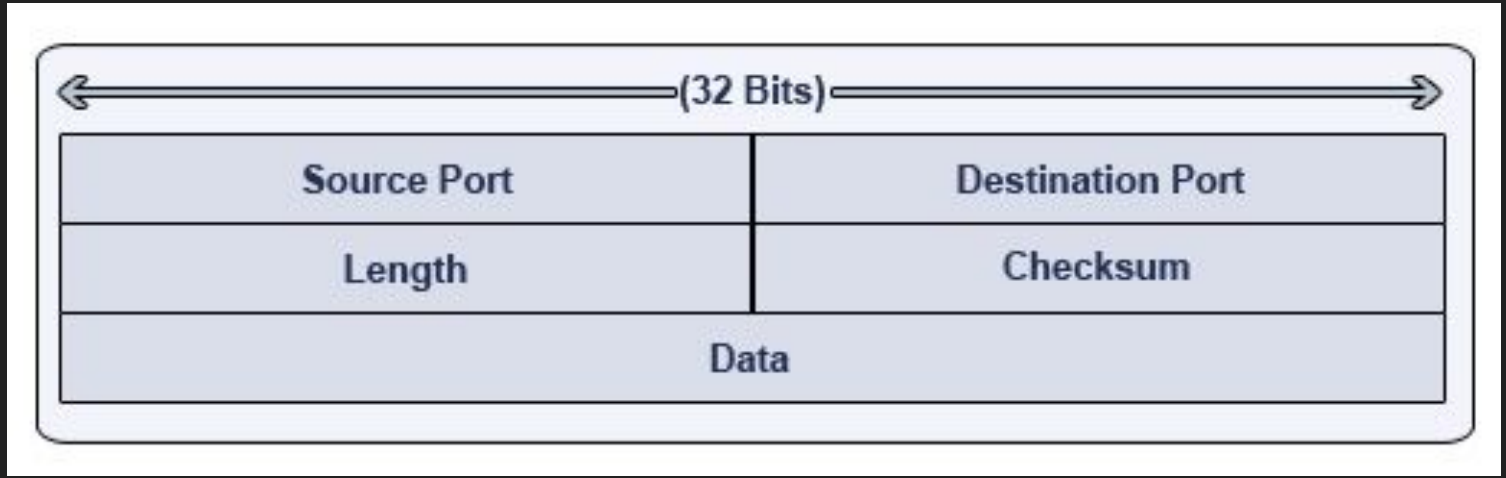
Корневые сертификаты и цепочки доверия



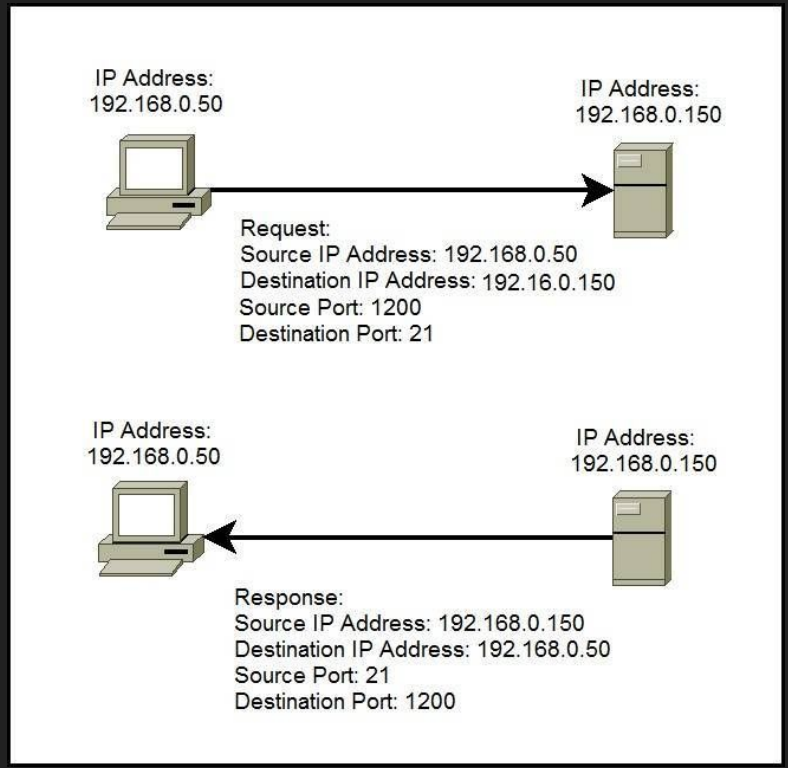
https://www.checktls.com/showcas.html

Мы можем отправлять данные на сервер.
Как взаимодействовать?

UDP – User Datagram Protocol



UDP – Получение ответа от сервера

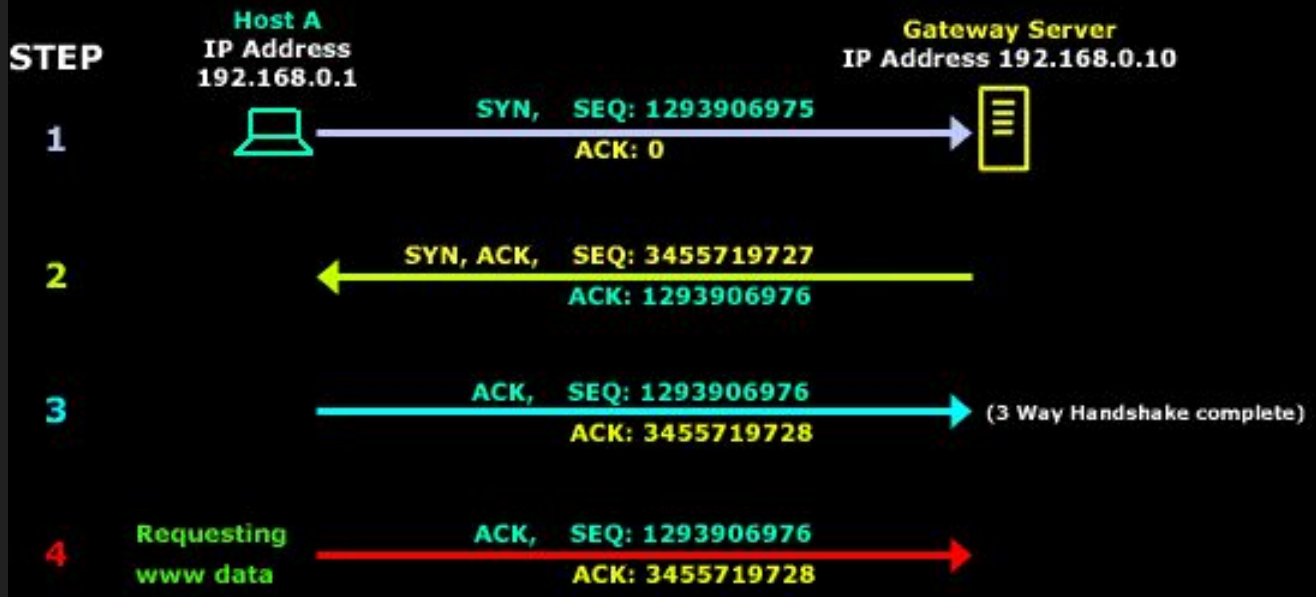


TCP – Transmission Control Protocol

TCP segment header

Offsets	Octet	0								1								2								3							
Octet	Bit	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0
0	0	Source port																Destination port															
4	32	Sequence number																															
8	64	Acknowledgment number (if ACK set)																															
12	96	Data offset				Reserved 0 0 0			NS	CWR	ECE	URG	ACK	PSH	RST	SYN	FIN	Window Size															
16	128	Checksum																Urgent pointer (if URG set)															
20 ...	160 ...	Options (if *data offset* > 5. Padded at the end with "0" bytes if necessary.) ...																															

Examining Sequence & Acknowledgment Numbers

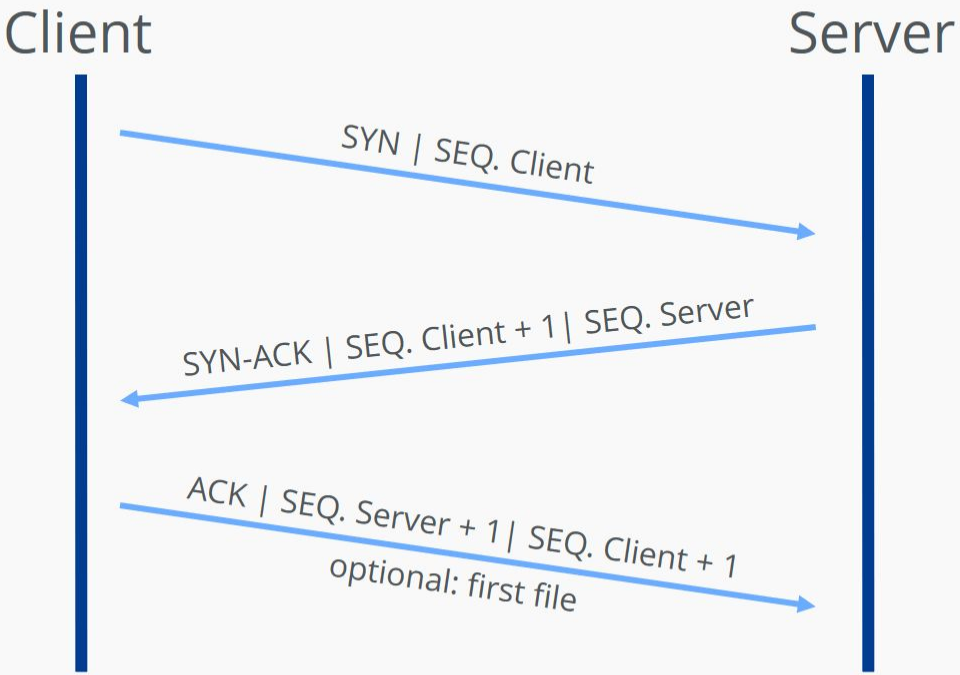


	Addr. IP src	Addr. IP dest	Protocol	Port src	Port dest	SEQ	ACK
1	192.168.0.1	192.168.0.10	TCP-> HTTP	3162	80	1293906975	0
2	192.168.0.10	192.168.0.1	TCP-> HTTP	80	3162	3455719727	1293906976
3	192.168.0.1	192.168.0.10	TCP-> HTTP	3162	80	1293906976	3455719728
4	192.168.0.1	192.168.0.10	TCP-> HTTP	3162	80	1293906976	3455719728

Here you can see how the Seq. & Ack. numbers are exchanged. The first 3 steps are to establish the connection and the data request (step 4) from the client follows

Концепция соединения

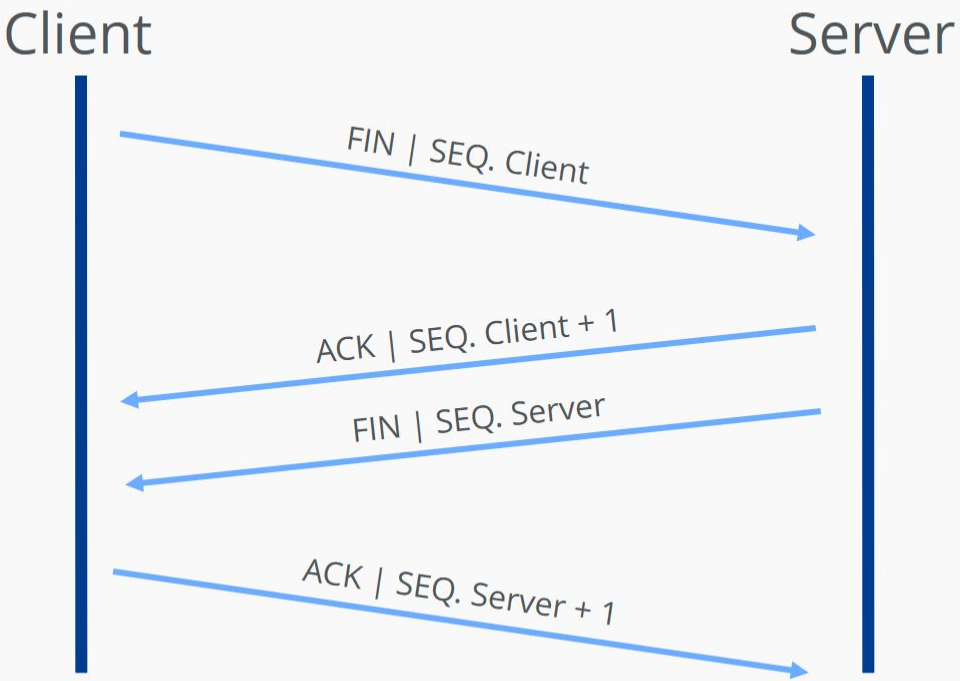
TCP connection establishment (Three way handshake)



SEQ. = Sequence number

IONOS

TCP connection termination (TCP Teardown)



SEQ. = Sequence number

IONOS

Флаги

SYN: Synchronize sequence numbers. Only the first packet sent from each end should have this flag set. Some other flags and fields change meaning based on this flag, and some are only valid when it is set, and others when it is clear.

ACK: Indicates that the Acknowledgment field is significant. All packets after the initial SYN packet sent by the client should have this flag set.

PSH: Push function. Asks to push the buffered data to the receiving application.

FIN: Last packet from sender

RST: Reset the connection

The rule of TCP is there is no rule of TCP.

Размер окна

- Получатель обозначает максимальный размер своего буфера на прием
- Отправитель обязан остановить отправку, если достигнут размер окна неподтвержденных данных:
- Получатель:
  - Отправляет ACK каждый пакет (RFC793)
  - Отправляет ACK каждый второй пакет (RFC1122)
  - Часто реализации отправляют после PSH (не всегда)
  - Часто реализации отправляют ACK по таймауту (не всегда)

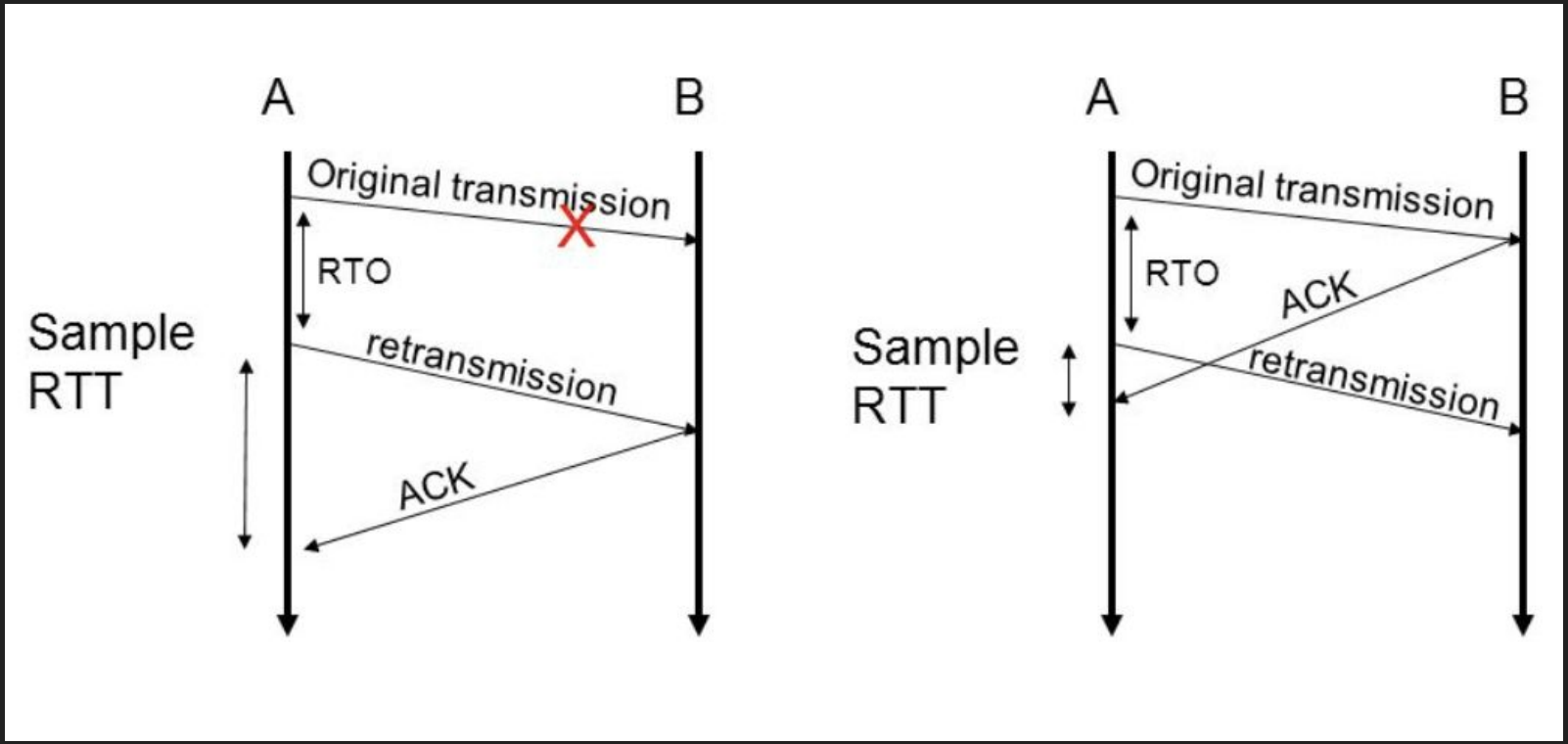
Опции

- Maximum Segment Size (MSS)
- Window Scaling
- Selective Acknowledgements (SACK)
- Timestamps
- Nop

Проблемы TCP

- Retransmission ambiguity
- Head of line blocking
- Ossification
- Handshake latency

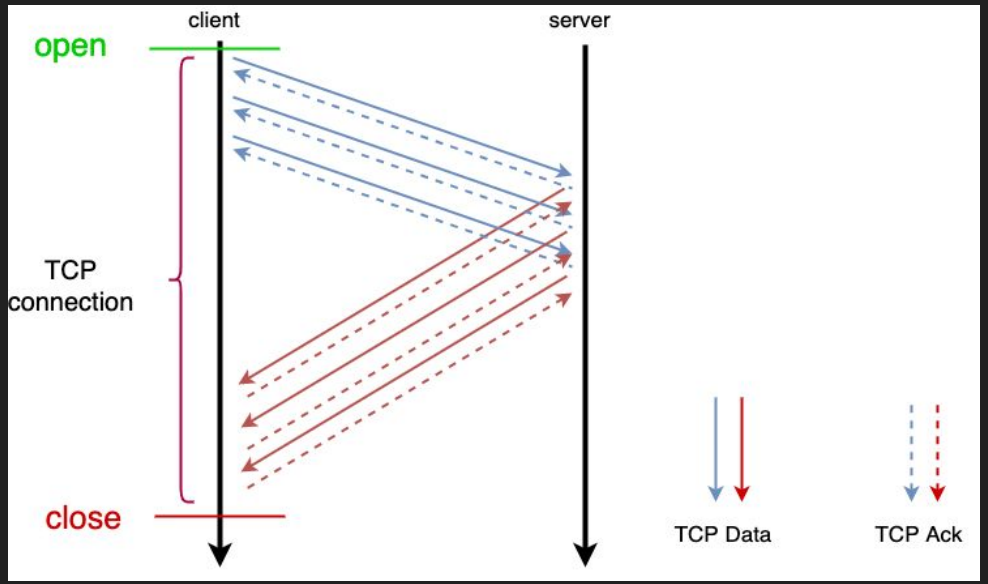
Retransmission ambiguity



Head of line (HOL) blocking

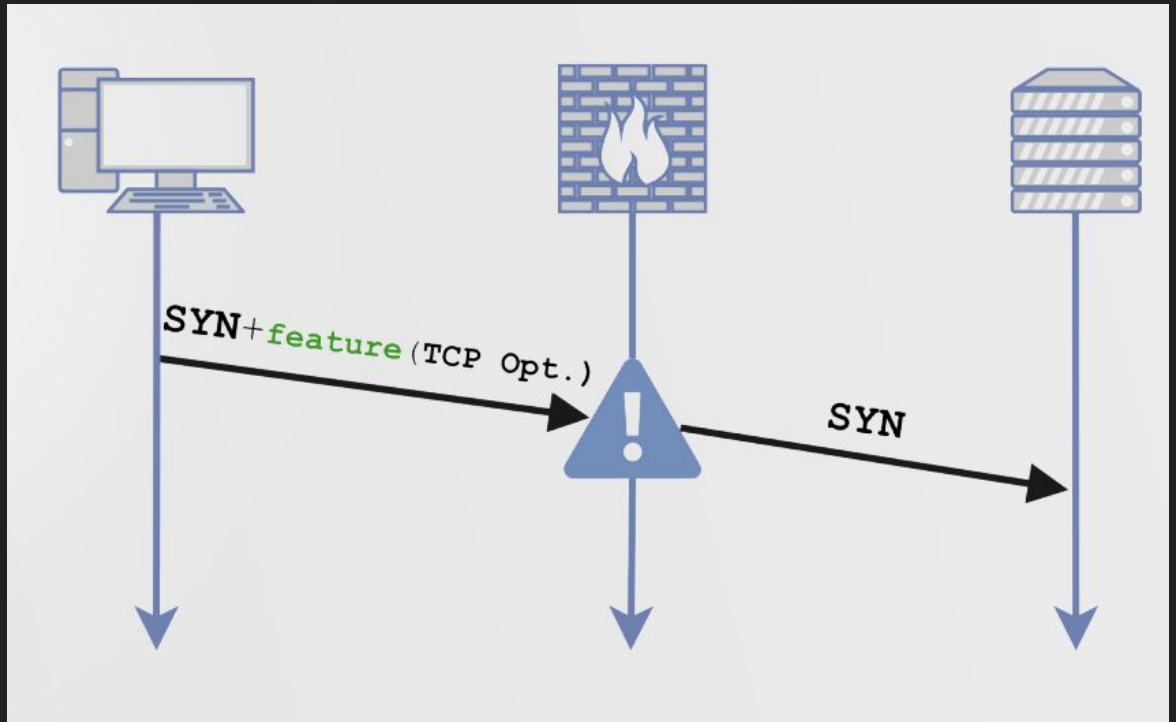
HTTP 1.1 - много соединений

HTTP 2 - мультиплексирование живет внутри одного TCP соединения



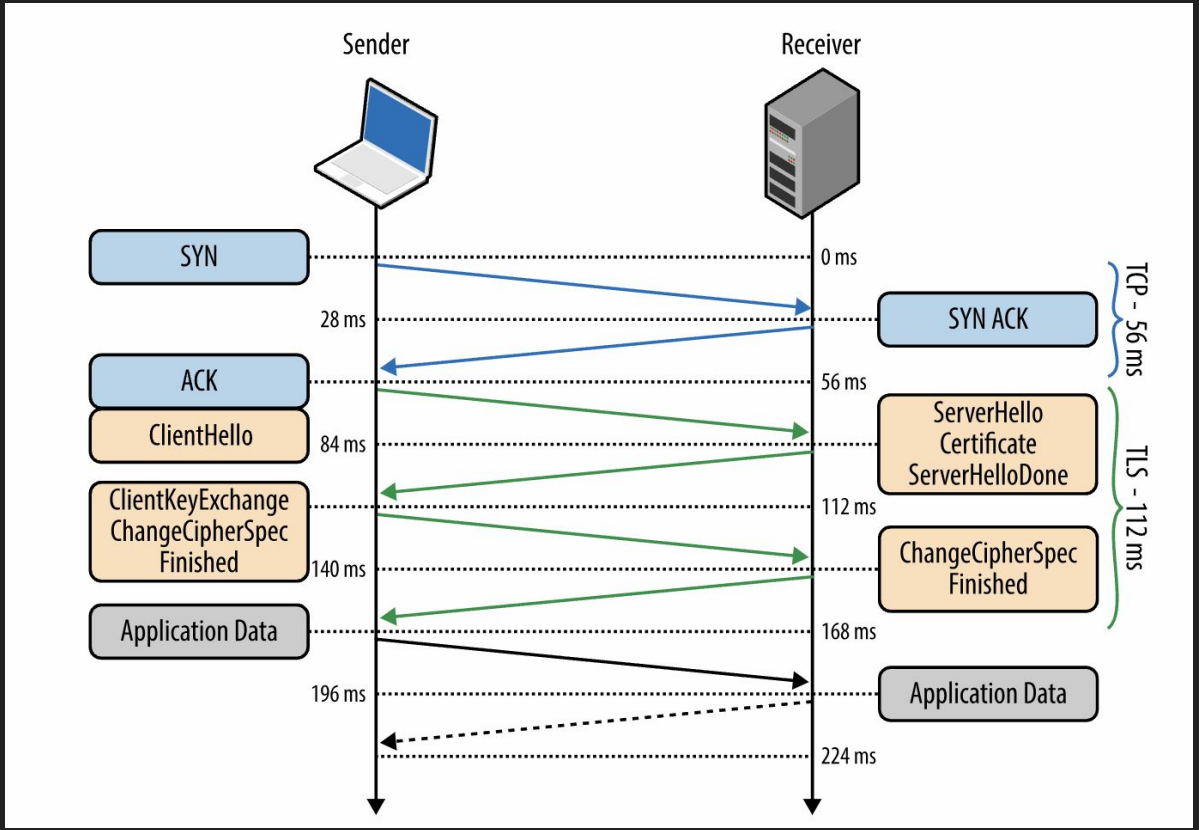
Ossification

Роутеры посередине парсят TCP, обновления протокола невозможны



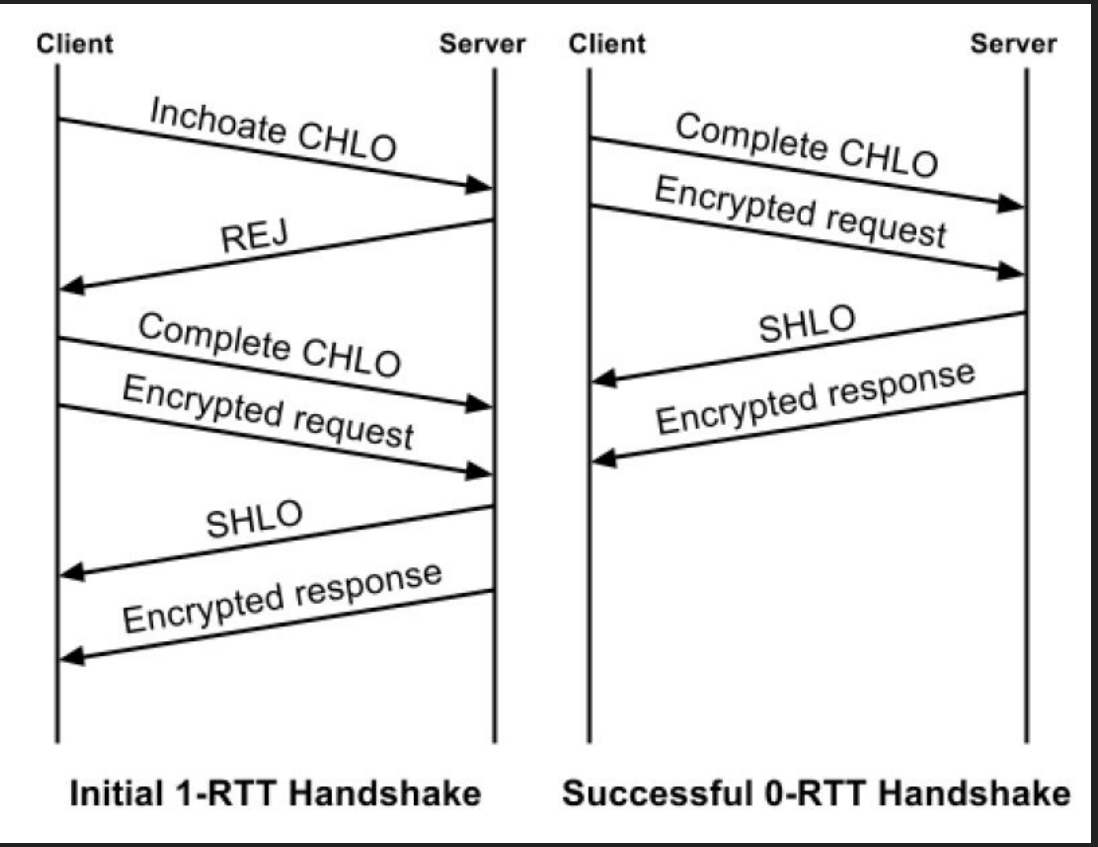
Handshake latency

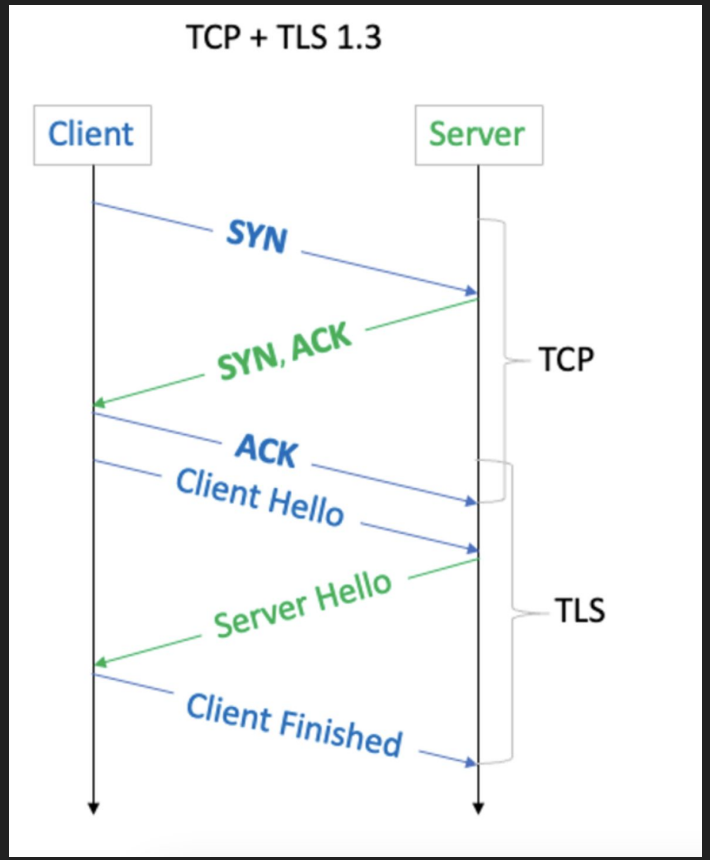
- TCP Handshake
- TLS Handshake



QUIC

QUIC - 0-rtt handshake vs TCP + TLS Handshake

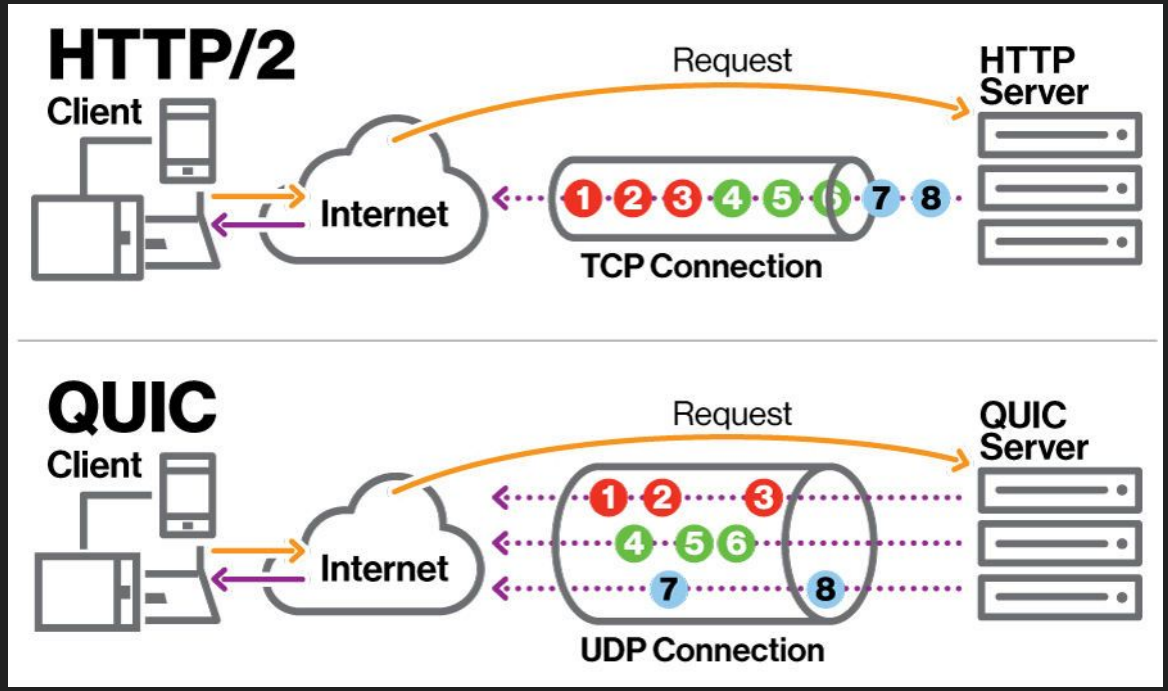




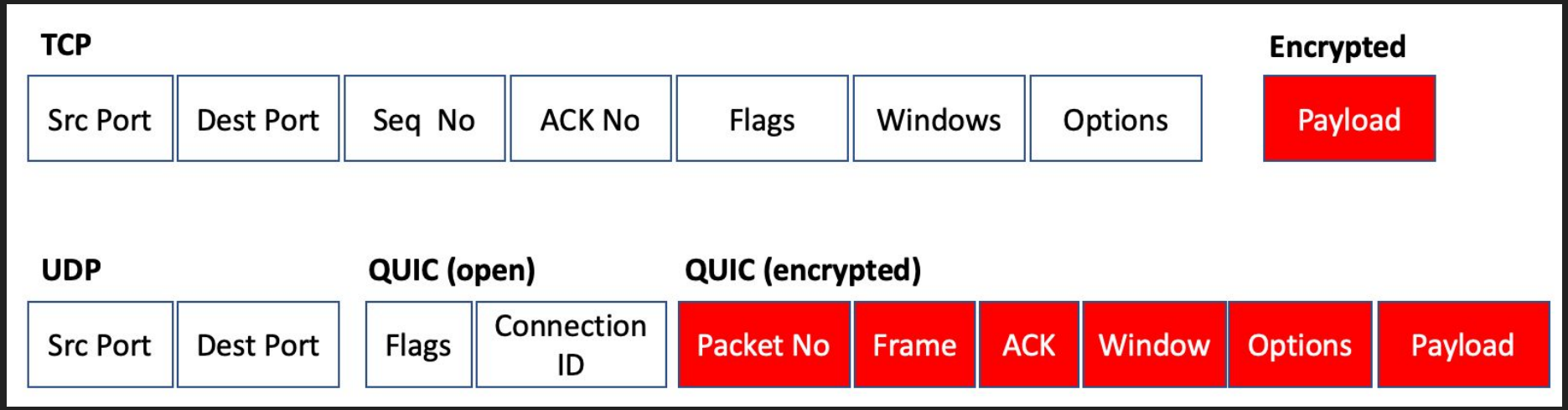
QUIC streams – Head of Line blocking

Чаще всего теряется всего один пакет.

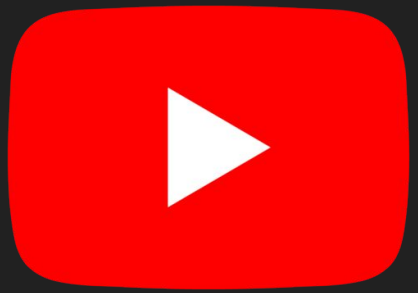
Connection ID



QUIC - Шифрование и заголовок против оссификации



Только Google могли придумать QUIC



		% latency reduction by percentile						
		Lower latency				Higher latency		
	Mean	1%	5%	10%	50%	90%	95%	99%
Search								
Desktop	8.0	0.4	1.3	1.4	1.5	5.8	10.3	16.7
Mobile	3.6	-0.6	-0.3	0.3	0.5	4.5	8.8	14.3
Video								
Desktop	8.0	1.2	3.1	3.3	4.6	8.4	9.0	10.6
Mobile	5.3	0.0	0.6	0.5	1.2	4.4	5.8	7.5

Преимущества QUIC

- Уменьшает задержку
- Решает проблему head-of-line blocking
- Можно обновлять

Проблемы QUIC

- Код в userspace - до 2x CPU по сравнению с TCP
- Не дает прироста в производительности в low-latency, low-loss сетях

Как прокладываются маршруты?

Interior Gateway Protocols

- Протоколы, которые работают внутри одной Автономной Системы (AS).
- Обычно знают топологию (карту) сети
- Выбираются на основе оптимальности: пропускной способности и задержки
- Отказоустойчивые: если падает ребро, перестраиваются и находят маршруты заново

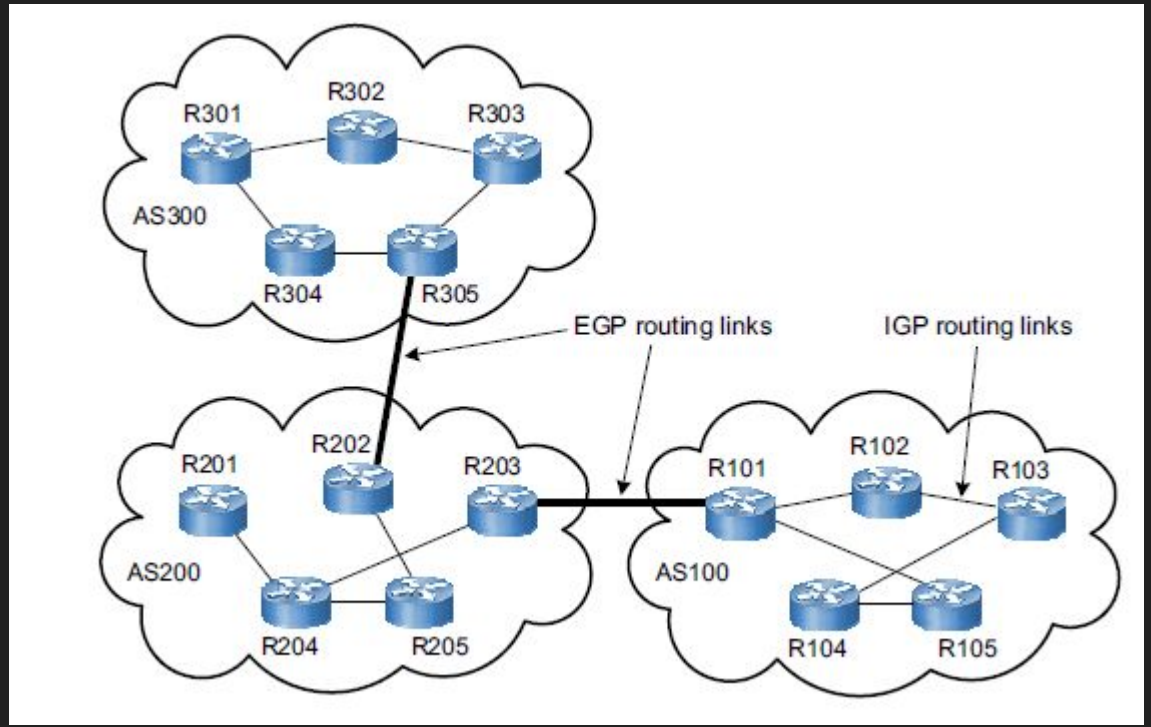
RIP: просто обмен маршрутами с соседями

OSPF: хранит всю сеть и ищет кратчайшие пути алгоритмом Дейкстры. Может делить сеть на зоны

Exterior Gateway Protocols

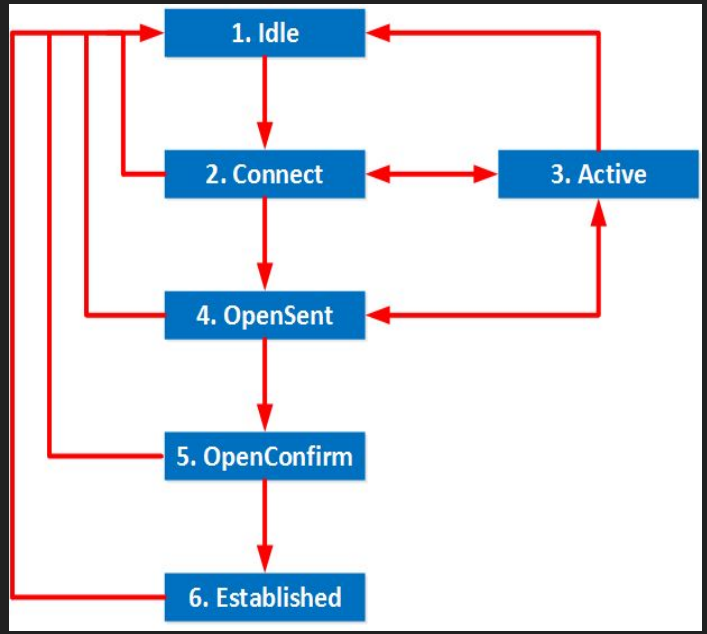
- Протоколы, которые соединяют между собой автономные зоны в большой сети
- Не знают внутреннюю структуру сетей
- Существует лишь одна по-настоящему большая сеть (и один протокол)

EGP vs IGP


R301
R302
R303
AS300
R304
R305
EGP routing links
IGP routing links
R201
R202
R203
AS200
R204
R205
R101
R102
R103
AS100
R104
R105

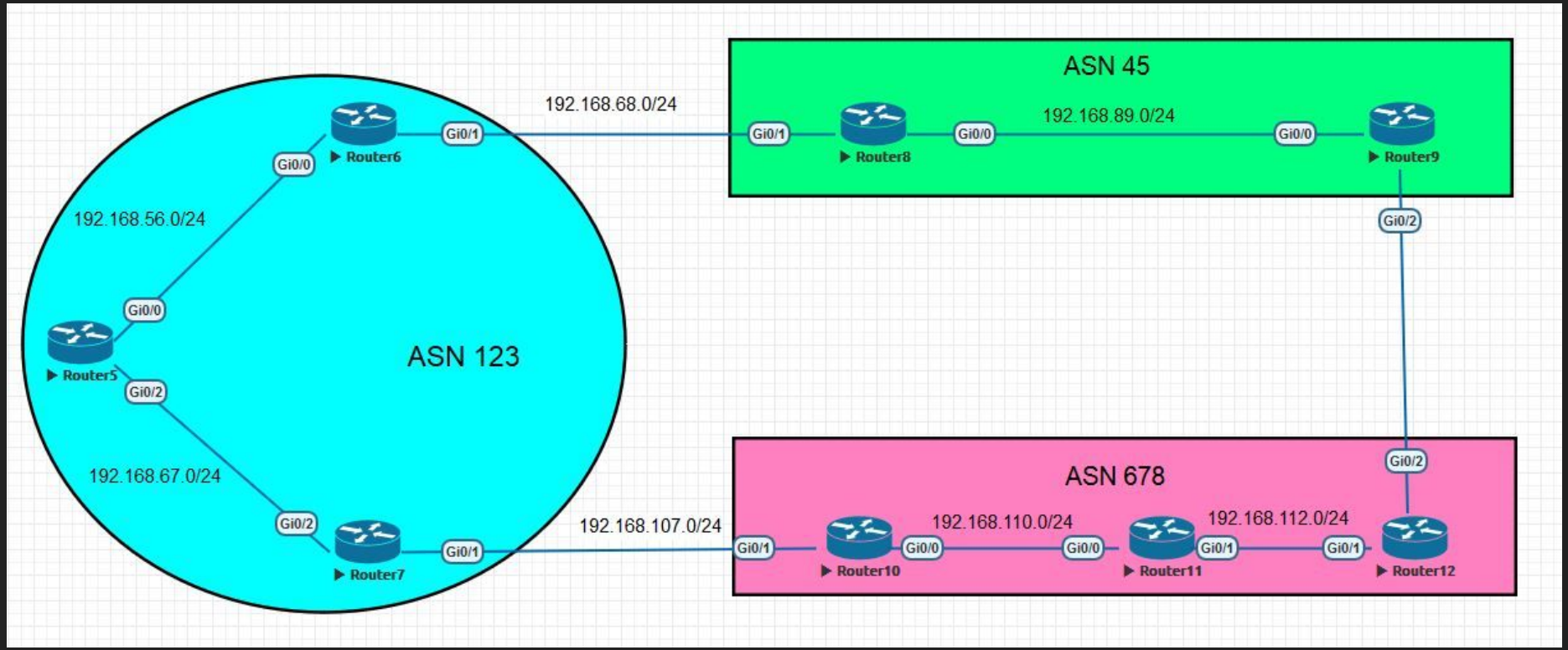
BGP – Border Gateway Protocol

- Соседство настраивается заранее, нет auto-discovery.
- Используется TCP для достижения консистентной и инкрементальной сессии.



BGP типы сообщений
Open
Keepalive
Update
Notification
*Route Refresh

AS - Autonomous System



Доверие, безопасность и инциденты

2004 - Турция

2008 - Пакистан и YouTube

2014 - Канадский криптомайнинг

Больше примеров https://en.wikipedia.org/wiki/BGP_hijacking