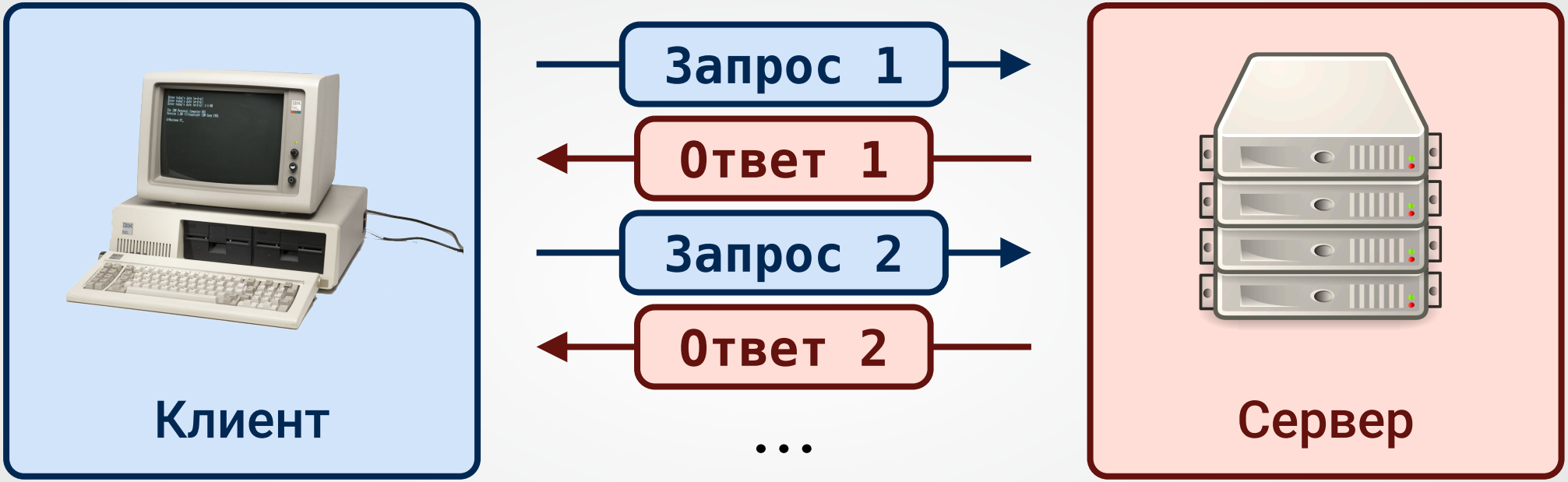
Сетевое взаимодействие 3

АКОС, МФТИ

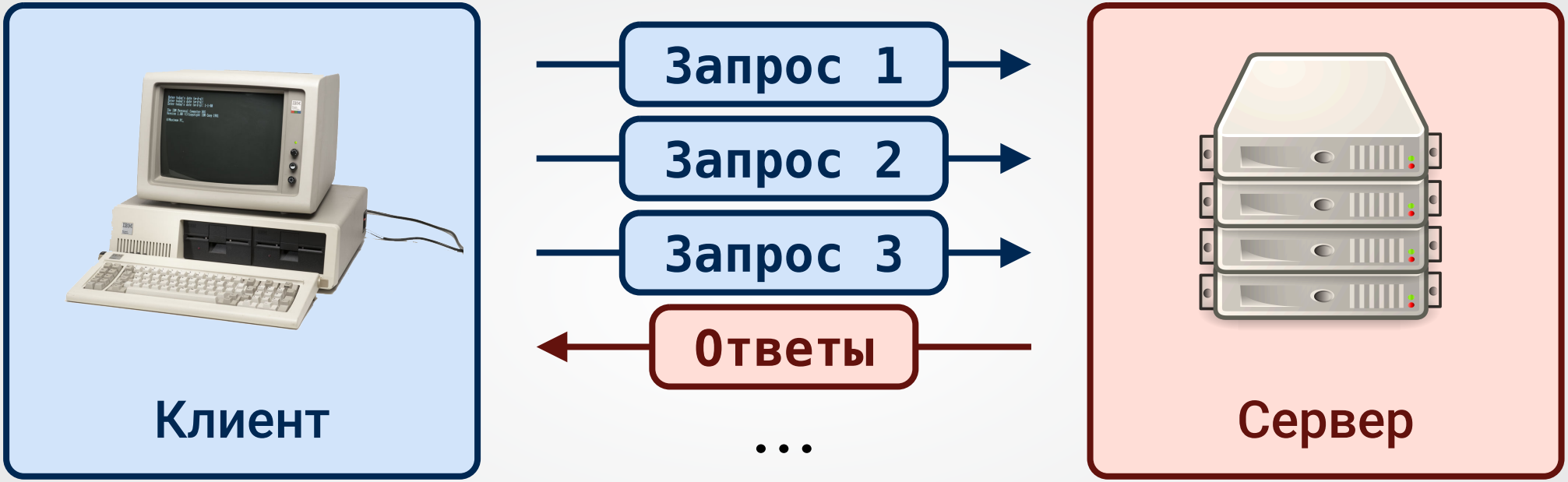
28 ноября, 2024

HTTP/1.1



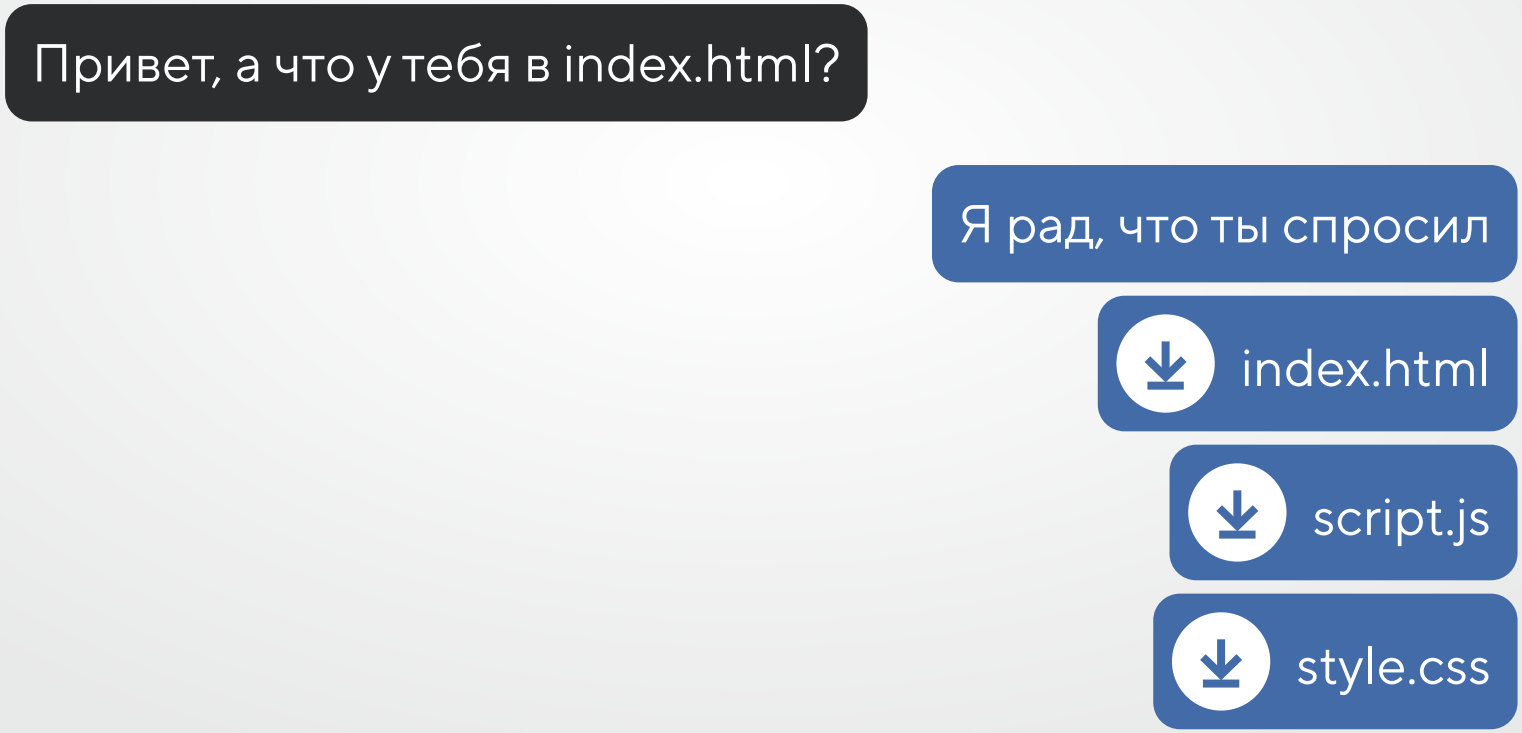
- **Запросы блокирующие:** сервер не может отвечать на следующий запрос, пока не ответил на текущий. По крайней мере, если соединение одно.

HTTP/2.0



- Мультиплексирует несколько запросов в рамках одного соединения.
- Клиент может отправить много запросов и задавать приоритет их обработки. (Например, сначала загрузить текст, а потом шрифты, картинки, и т.д)

- Обычно сервер примерно знает, какие ресурсы запросит клиент в будущем. Отправляя HTML-страницу, можно сразу начать отправлять картинки, стили, и скрипты, которые на ней есть.
- Это называется **Server Push**, и это тоже фича HTTP/2.0.



Что ещё поменяли в HTTP/2.0?

- **Протокол стал бинарным.** Напомним, HTTP/1.1 был текстовым.
- **Сжимаются заголовки** в запросах и ответах.
- Браузеры поддерживают его только как HTTPS, хотя стандарт этого не требует.

Шифрование

Зачем вообще нужно шифрование?

- Ваши сообщения в канале (Wi-Fi, Ethernet, …) могут подслушивать.
- Ваши сообщения могут быть изменены кем-то по пути.
- Кто угодно может выдать своё сообщение за ваше.

Алиса
Отправитель

Ева
Злостный злот

Боб
Получатель

Проблема передачи ключа

- Допустим, мы умеем шифровать сообщение каким-то паролем (ключом).
- Как Алисе и Бобу договориться о пароле так, чтобы Ева его не узнала?
- Передать пароль в открытую нельзя, иначе Ева его подслушает.

Проблема передачи ключа

- Допустим, мы умеем шифровать сообщение каким-то паролем (ключом).
- Как Алисе и Бобу договориться о пароле так, чтобы Ева его не узнала?
- Передать пароль в открытую нельзя, иначе Ева его подслушает.
- Для этого существует **алгоритм Диффи-Хеллмана**.

Алгоритм Диффи-Хеллмана

P, G можно передать в открытую

P – простое, G – его первообразный корень

```
1  a = rand() % P;
2  A = pow(G, a) % P;
3  send(A);
4  receive(&B);
5  K = pow(B, a) % P;
```

```
1  b = rand() % P;
2  B = pow(G, b) % P;
3  send(B);
4  receive(&A);
5  K = pow(A, b) % P;
```

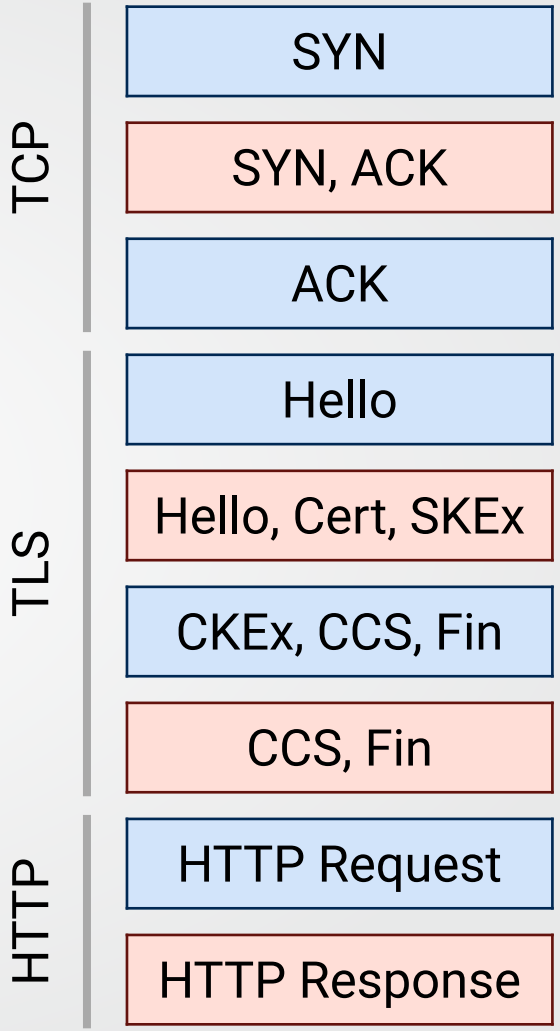
K - секретный ключ. Ева не сможет его узнать, даже если подслушает A, B, P и G.

TLS (Transport Layer Security)

- Защитный протокол транспортного уровня, обеспечивающий шифрование и связанные с ним гарантии.
- Использует алгоритм Диффи-Хеллмана для создания сеансового ключа.
- Шифрует сообщения, используя этот ключ.
- Позволяет проверить, что сервер – тот, за кого себя выдаёт.
- Позволяет проверить, что сообщение не было изменено по пути.
- Установка TLS-сеанса производится отдельным хендшейком длиной в 4 раунда.

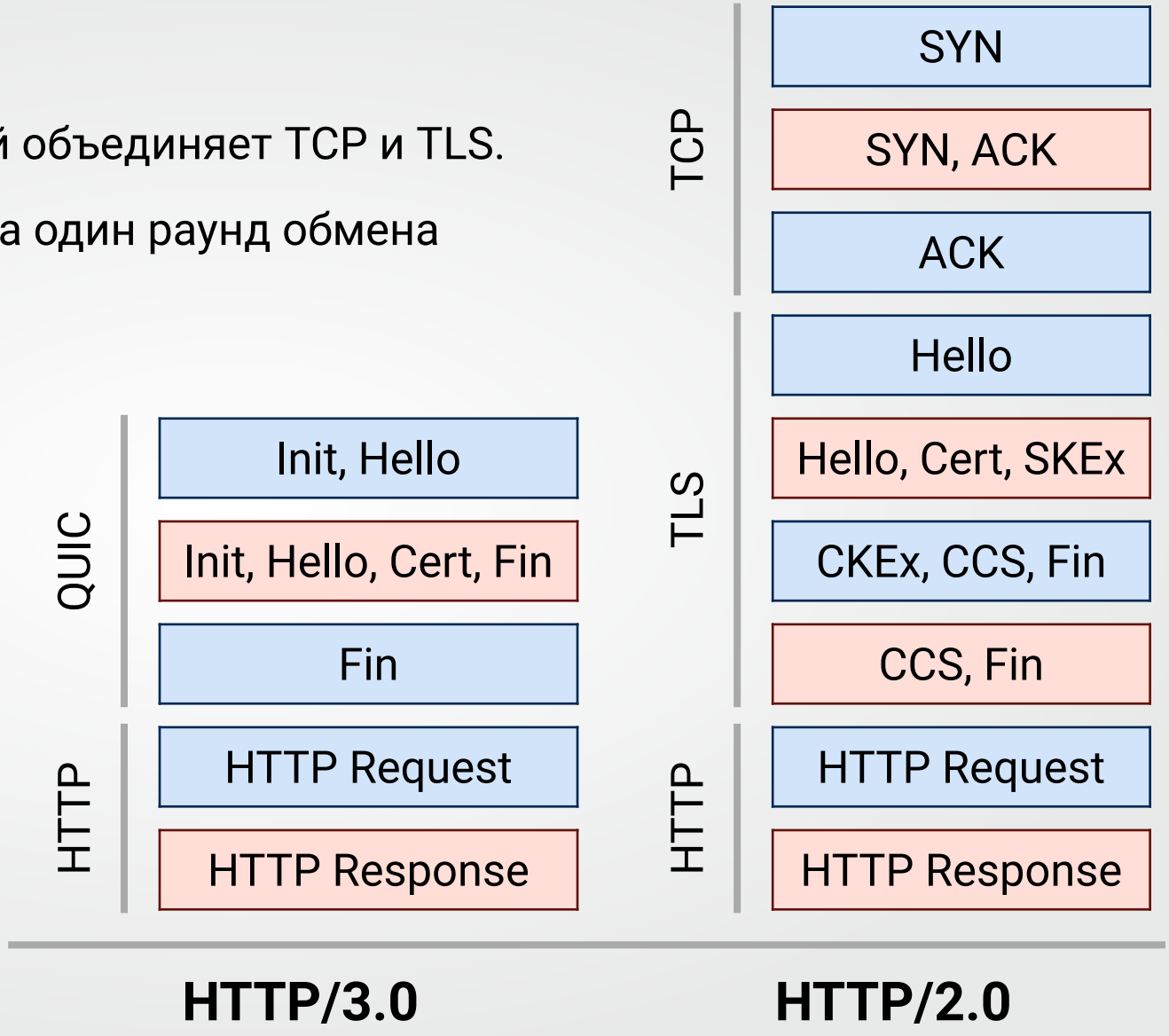
HTTPS, или бутерброд из хендшейков

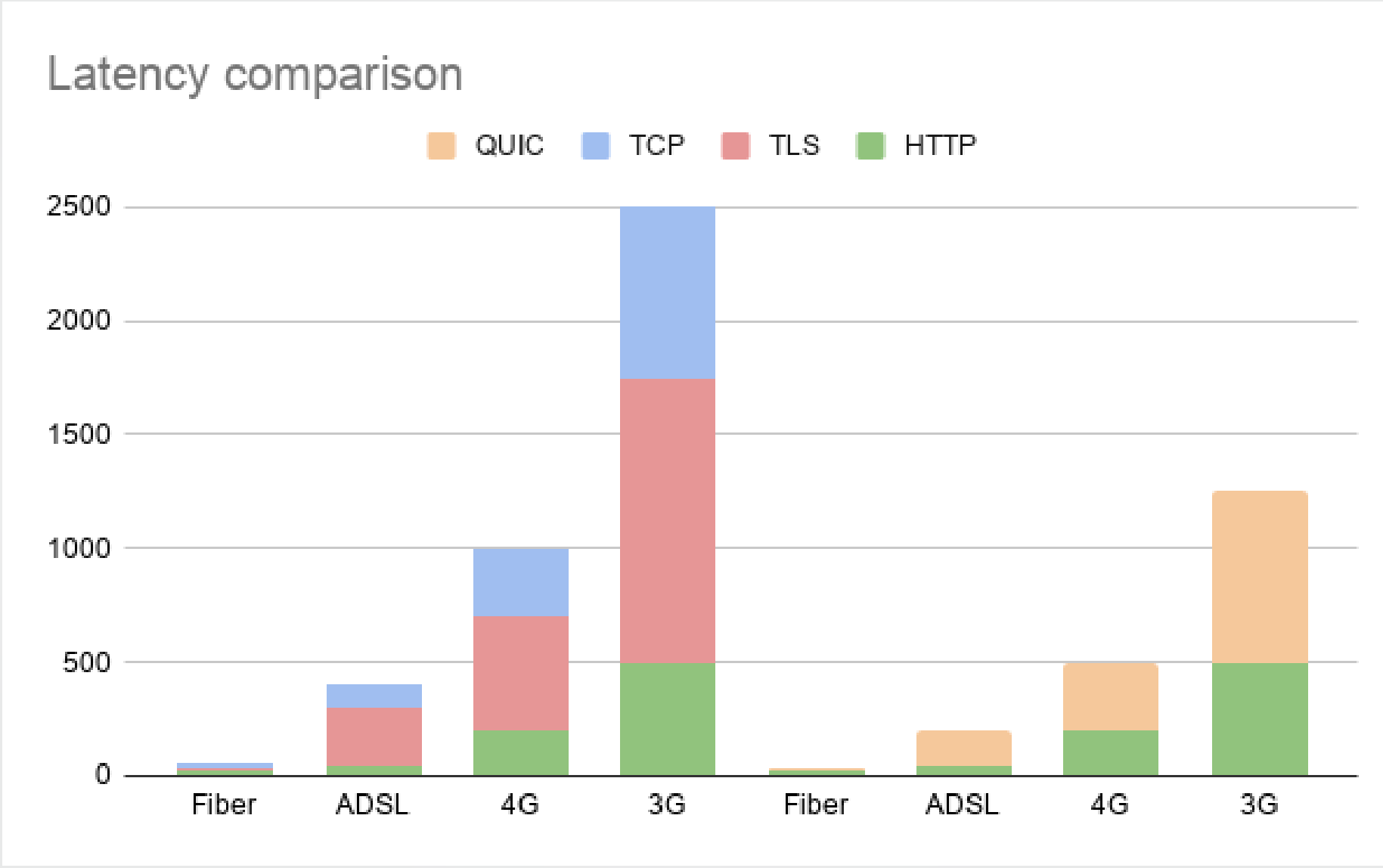
- Оригинальный HTTPS это HTTP over TLS. То есть между установкой TCP-соединения и передачей HTTP-запроса происходит установка сеанса TLS.
- Разделение обязанностей между TCP и TLS приводит к тому, что нужно совершать два хендшейка.
- Получается **семь** раундов обмена данными, чтобы загрузить страницу.
- В HTTP/3.0 цепочку TCP + TLS заменили на протокол QUIC. Он умеет делать оба хендшейка за один раунд обмена данными.
- HTTP/3.0 требует всего два раунда обмена вместо семи.



HTTP/3.0 и QUIC

- QUIC – это протокол, который объединяет TCP и TLS.
- Хендшейк QUIC происходит за один раунд обмена данными.
- Кроме этого QUIC позволяет делать **асинхронное мультиплексирование** запросов.
- [Статья от VK]




Latency comparison
QUIC
TCP
TLS
HTTP
2500
2000
1500
1000
500
0
Fiber
ADSL
4G
3G
Fiber
ADSL
4G
3G

Переезд в Userspace

- QUIC работает поверх UDP и реализован в пространстве пользователя.
- За счёт этого приложение может донастроить протокол под себя.
- Переход в Userspace - это общий тренд в сетевых технологиях.

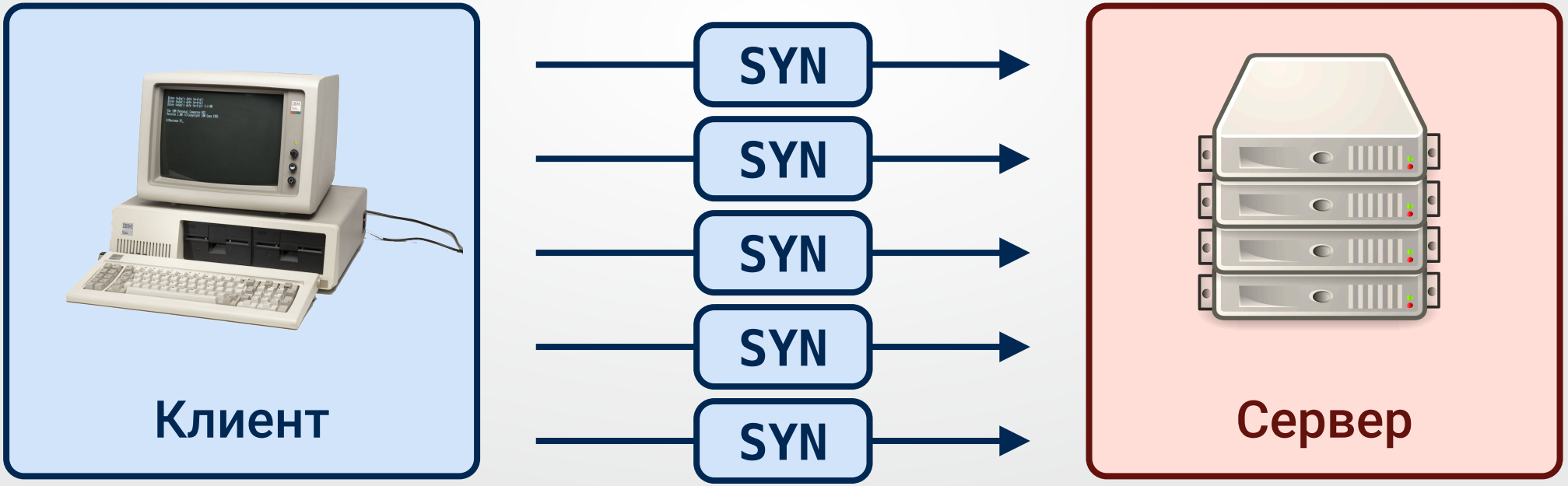
DPDK (Data Plane Development Kit)

- Переносит работу с сетью полностью в пространство пользователя.
- Использует DMA (Direct Memory Access) для обращения к буферам сетевой платы.
- Позволяет отправлять и принимать пакеты без переключений контекста.
- Отправить или принять пакет можно за 80 тактов процессора.
- Для сравнения, переключение контекста может занять 1000 тактов.

Давайте что-нибудь сломаем

Syn Flood

- Denial-Of-Service (DoS)-Атака на TCP-сервер.
- Атакующий отправляет много пакетов с флагом SYN и даже не ждёт ответа.
- Сервер должен создать на каждый пакет новое соединение и ждать таймаута.
- Очередь подключений переполняется и обычный клиент не может подключиться.



GET Flood / POST Flood

- Атака на HTTP-сервер. Атакующий отправлет много тяжелых запросов на сервер, чтобы перегрузить его. Для такой атаки нужно подобрать запрос, который будет занимать много ресурсов сервера.
- От таких атак защищаются кешированием, капчей или CDN (Например, Cloudflare).

Cache Bypassing

- То же, что и GET / POST Flood, но запросы подбираются таким образом, чтобы сервер не мог их кешировать.

Reverse Bandwidth Attack

- Перегрузка исходящего канала сервера. Например, много параллельных запросов на загрузку большого файла.

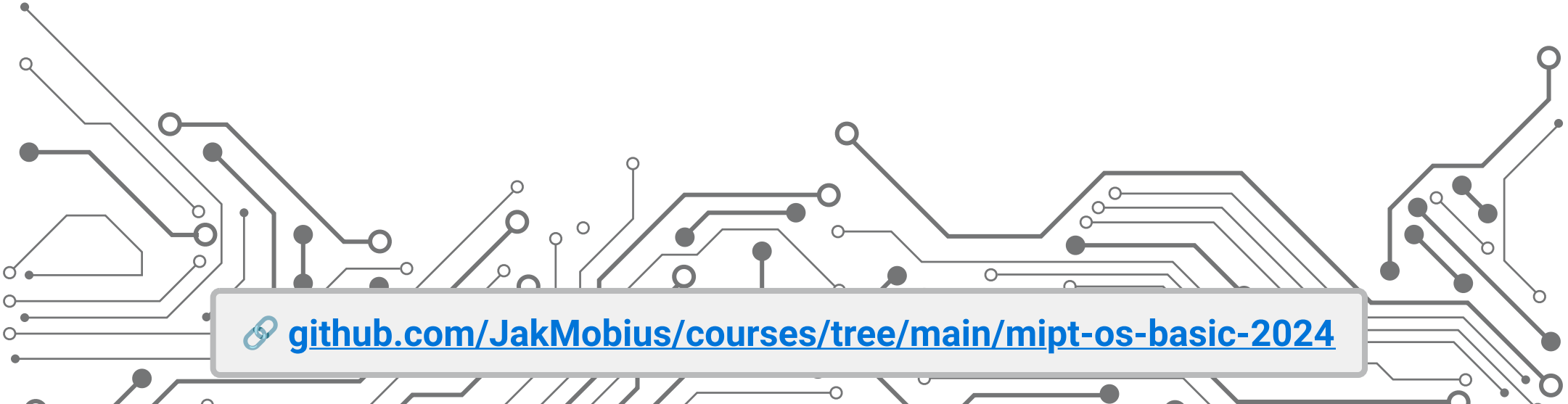
Low And Slow

- Перегрузка сервера большим количеством медленных запросов. Если сервер создаёт по потоку или по процессу на каждое соединение, ему будет особенно больно.

Атаки на DNS

- В 2016 году неизвестные хакеры атаковали DNS-сервер Dyn, который обслуживал многие крупные сервисы.
- У них был большой ботнет из умных чайников, камер, и других IoT-устройств. Запросы отправлялись с нескольких десятков миллионов IP-адресов.
- Пострадали не меньше 70 крупных сервисов.
- Владельца сайта от атаки на DNS не спасёт никакая DDoS-защита.
- В целом, можно заставить клиентов запомнить ваш IP-адрес. Ну, помним же мы номера телефонов…

Спасибо за внимание!



🔗 github.com/JakMobius/courses/tree/main/mipt-os-basic-2024