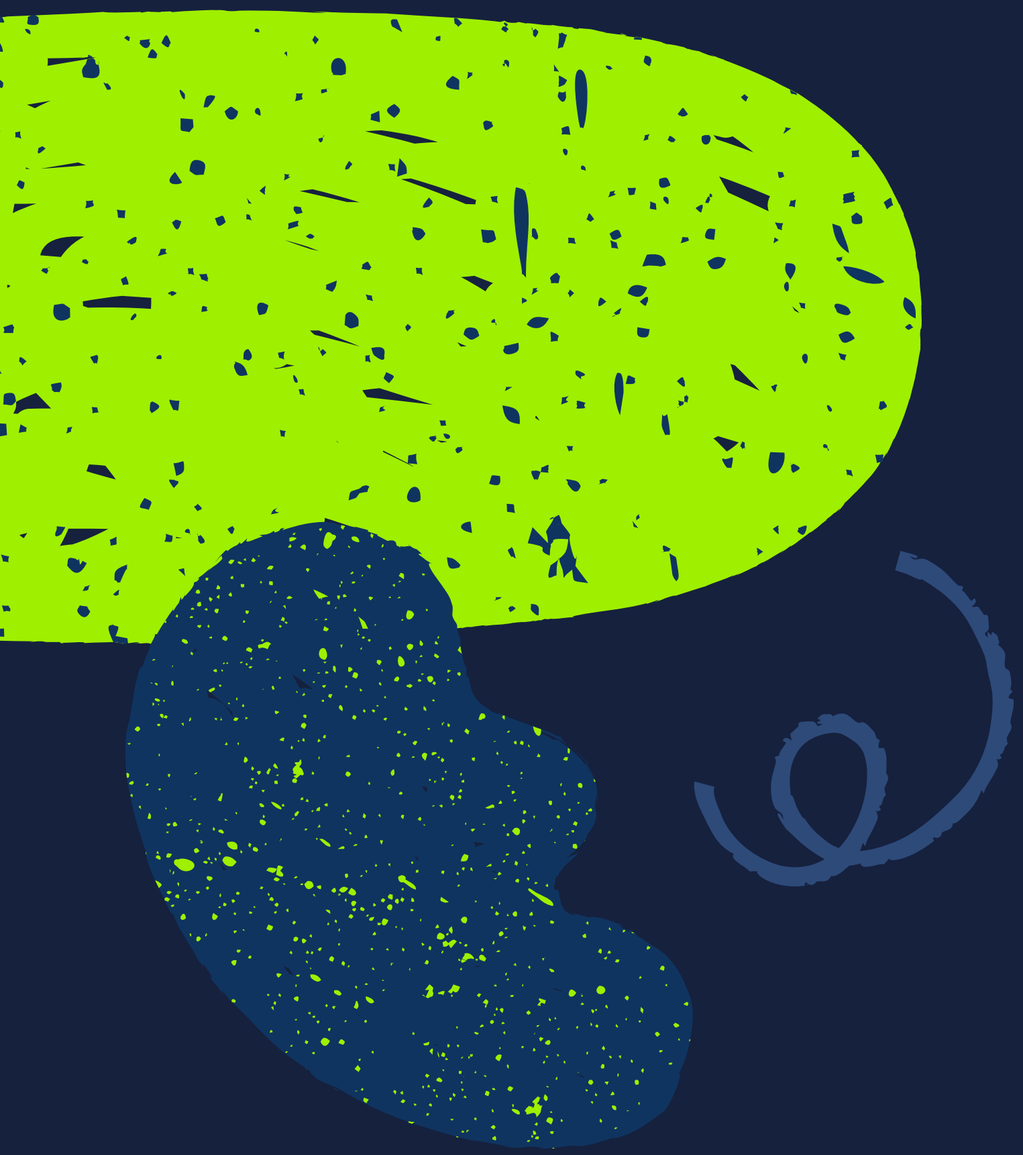
Core Technologies of the Web

Overview of Web Services Concepts

Ph.D Mohammed Sahraoui



Content

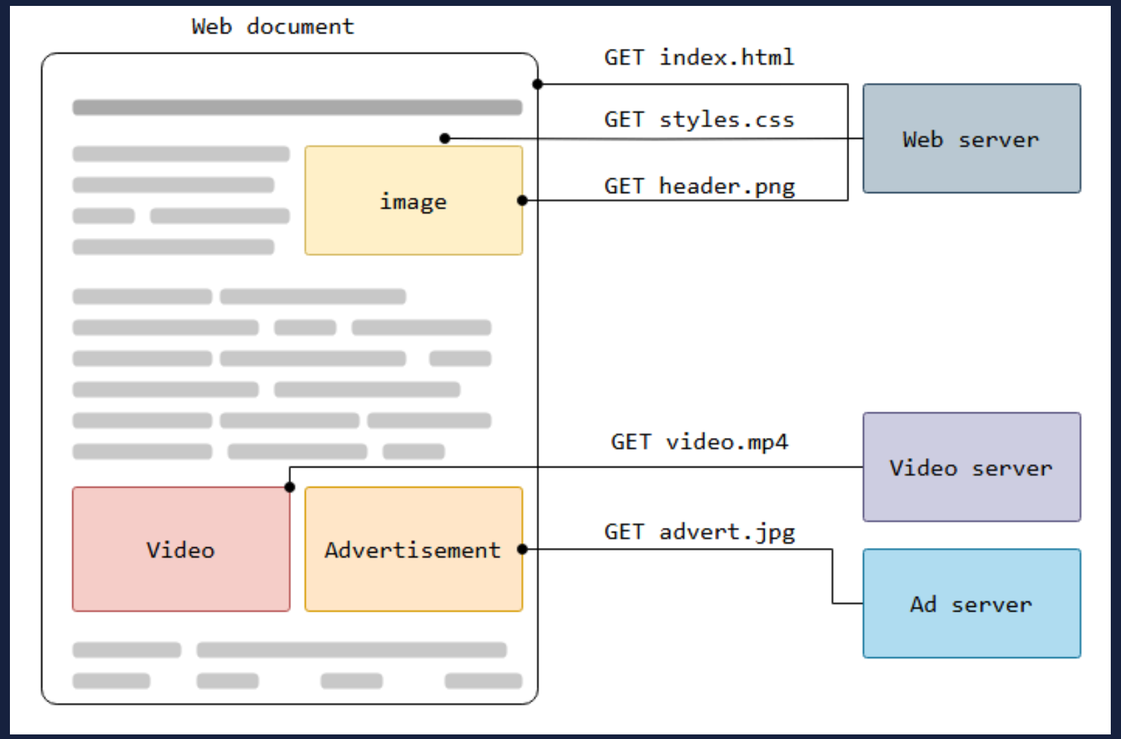
Chapter 2:

- 2.1 HTTP/HTTPS Deep Dive
  - HTTP Requests (Methods: GET, POST, PUT, DELETE, etc.)
  - HTTP Responses (Status Codes: 200, 201, 400, 401, 403, 404, 500)
  - HTTP Headers (Content-Type, Authorization, Cache-Control, CORS)
- 2.2 Data Formats for Exchange
  - XML (eXtensible Markup Language)
    - Structure, Elements, Attributes.
  - JSON (JavaScript Object Notation)
    - Syntax, Data Types, Advantages over XML.
- 2.3 Introduction to TLS/SSL (The "S" in HTTPS)
  - Purpose: Encryption, Authentication, Data Integrity.
  - High-level overview of how a TLS handshake works.

Ph.D Mohammed Sahraoui

HTTP/HTTPS Protocol

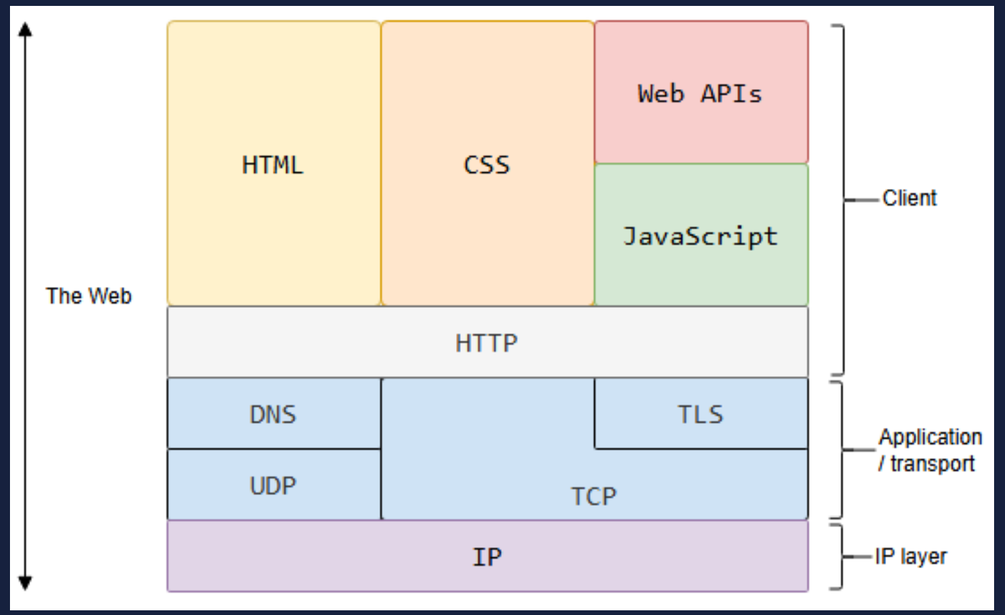
HTTP is a protocol for fetching resources such as HTML documents. It is the **foundation of any data exchange** on the Web and it is a client-server protocol, which means requests are initiated by the recipient, usually the Web browser. A complete document is typically constructed from resources such as text content, layout instructions, images, videos, scripts, and more.



https://developer.mozilla.org/

Clients and servers communicate by exchanging individual messages (as opposed to a stream of data). The messages sent by the client are called **requests** and the messages sent by the server as an answer are called **responses**.

Ph.D Mohammed Sahraoui



https://developer.mozilla.org/

Designed in the early 1990s, HTTP is an extensible protocol which has evolved over time. **It is an application layer** protocol that is sent over **TCP**, or over a **TLS-encrypted TCP connection**, though any reliable transport protocol could theoretically be used. Due to its extensibility, it is used to not only fetch hypertext documents, but also **images** and **videos** or to post content to servers, like with HTML form results. HTTP can also be used to fetch parts of documents to update Web pages on demand.

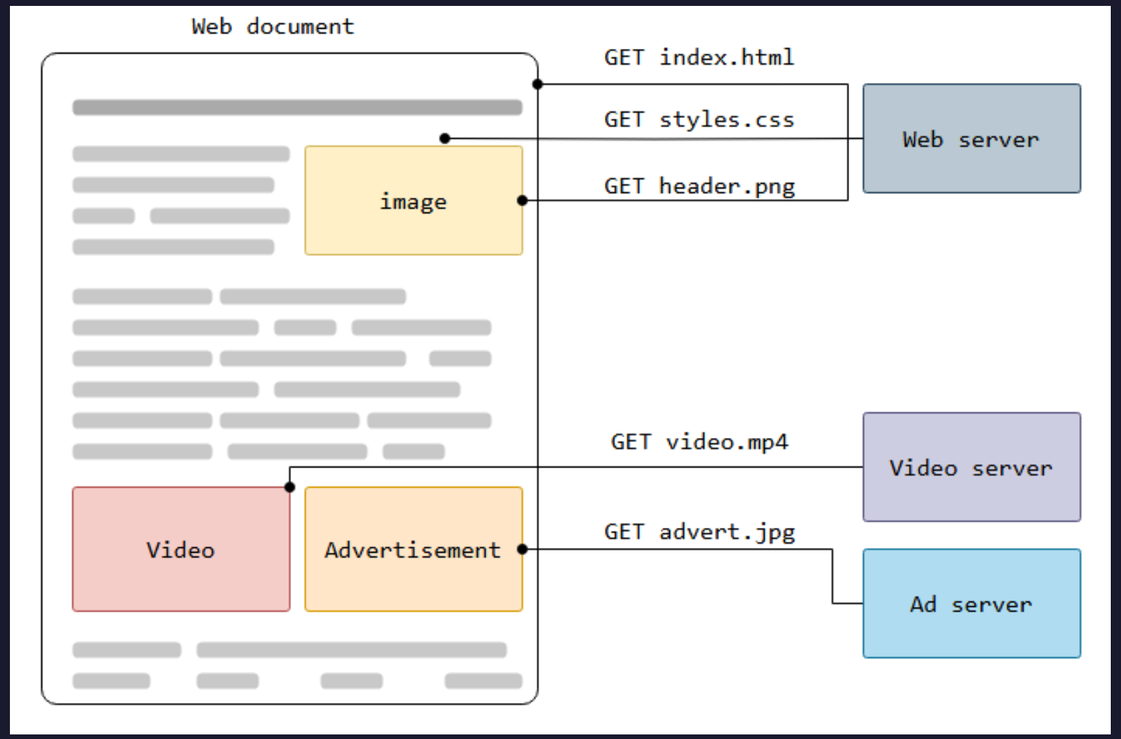

GET / HTTP 1.1

200 OK


200 OK

Components of HTTP-based systems

HTTP is a protocol for fetching resources such as HTML documents. It is the **foundation of any data exchange** on the Web and it is a client-server protocol, which means requests are initiated by the recipient, usually the Web browser. A complete document is typically constructed from resources such as text content, layout instructions, images, videos, scripts, and more.



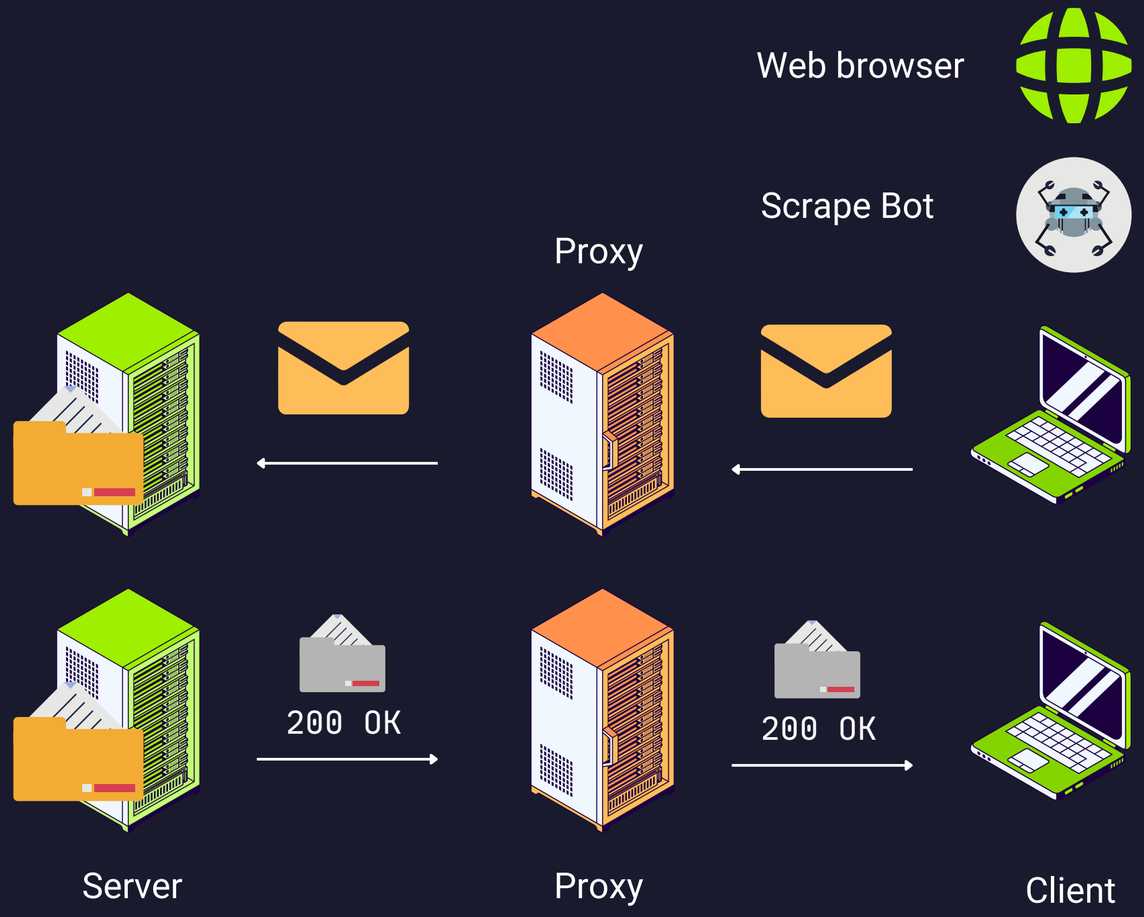
https://developer.mozilla.org/

Clients and servers communicate by exchanging individual messages (as opposed to a stream of data). The messages sent by the client are called **requests** and the messages sent by the server as an answer are called **responses**.

Ph.D Mohammed Sahraoui

Components of HTTP-based systems

HTTP is a client-server protocol: requests are sent by one entity, the user-agent (or a proxy on behalf of it). Most of the time the user-agent is a Web browser, but it can be anything, for example, a robot that crawls the Web to populate and maintain a search engine index. Each individual request is sent to a server, which handles it and provides an answer called the response. Between the client and the server there are numerous entities, collectively called **proxies**, which perform different operations and act as gateways or **caches**, for example.



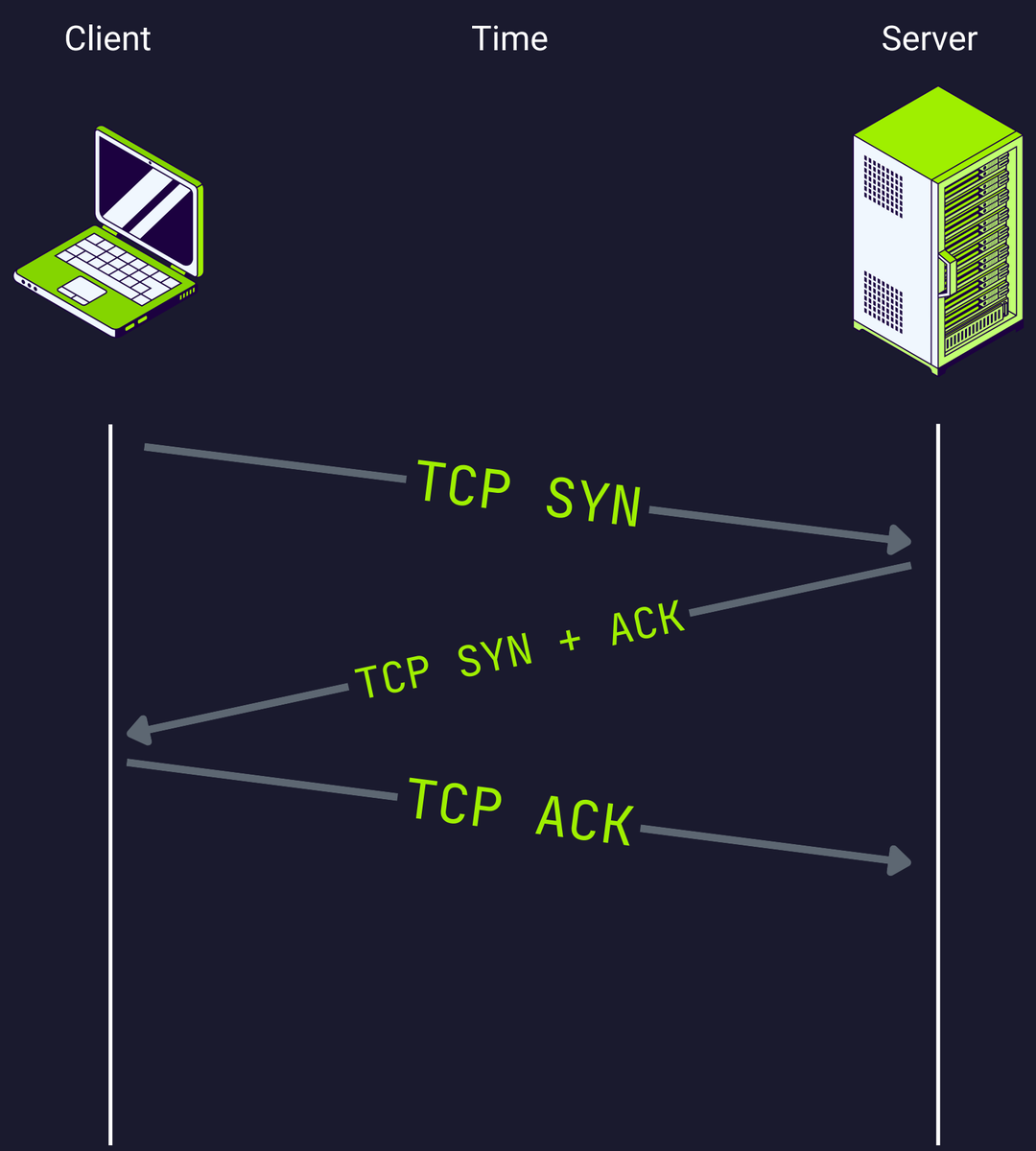
The Web view is abstract from the network infrastructure.

Ph.D Mohammed Sahraoui

Basic aspects of HTTP

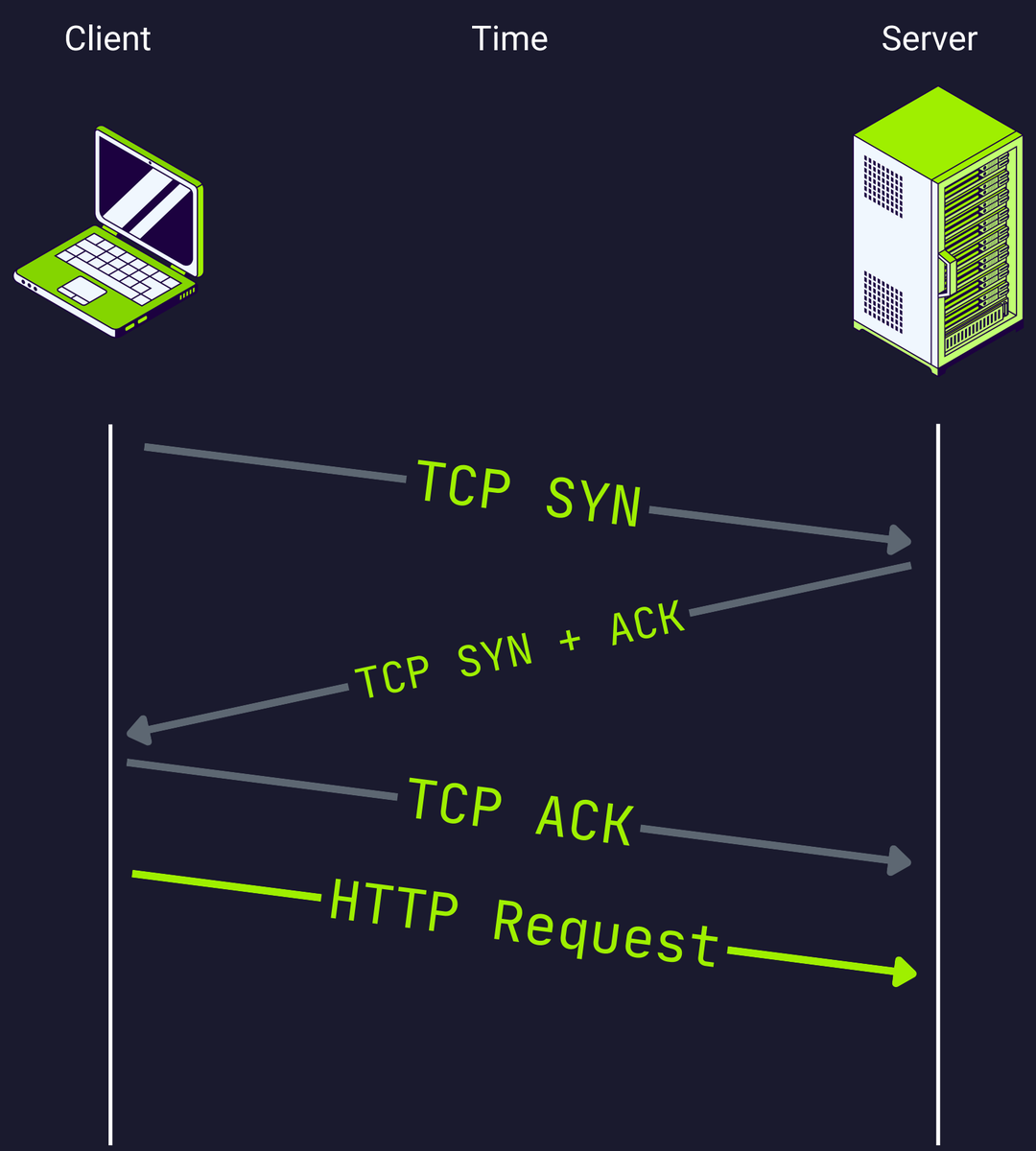
Version	Year Introduced	Key Features
HTTP/1.0	1996	Simple request/response model , No persistent connections (each request opens a new TCP connection)
HTTP/1.1	1997	Persistent connections via keep-alive , Chunked transfer encoding, Pipelining (limited adoption)
HTTP/2	2015	Binary framing (not text-based), Multiplexing (multiple requests over one connection), Header compression (HPACK)
HTTP/3	2022	Based on QUIC (UDP instead of TCP) , Faster connection setup - Improved reliability and security

HTTP 1.0



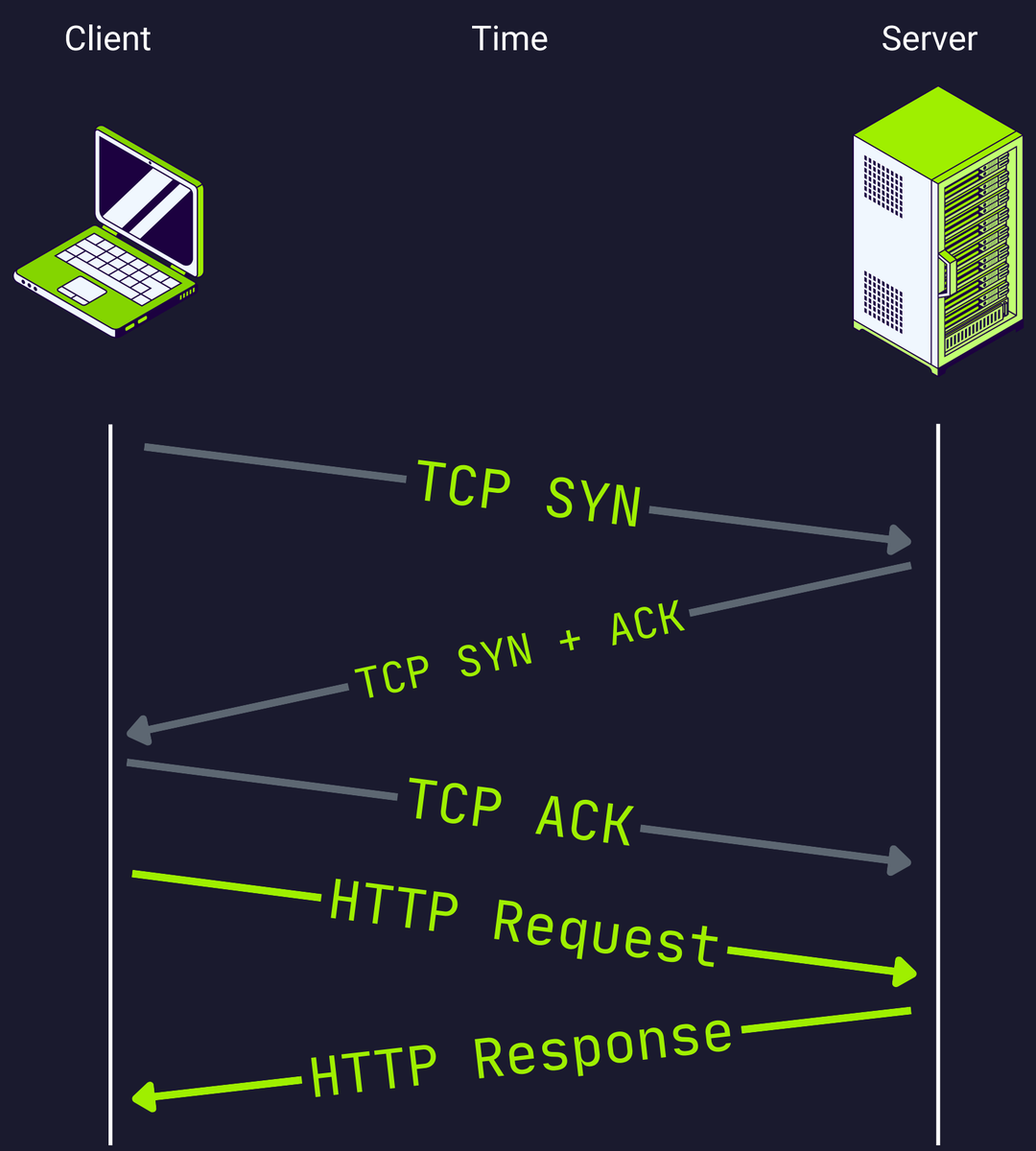
Ph.D Mohammed Sahraoui

HTTP 1.0



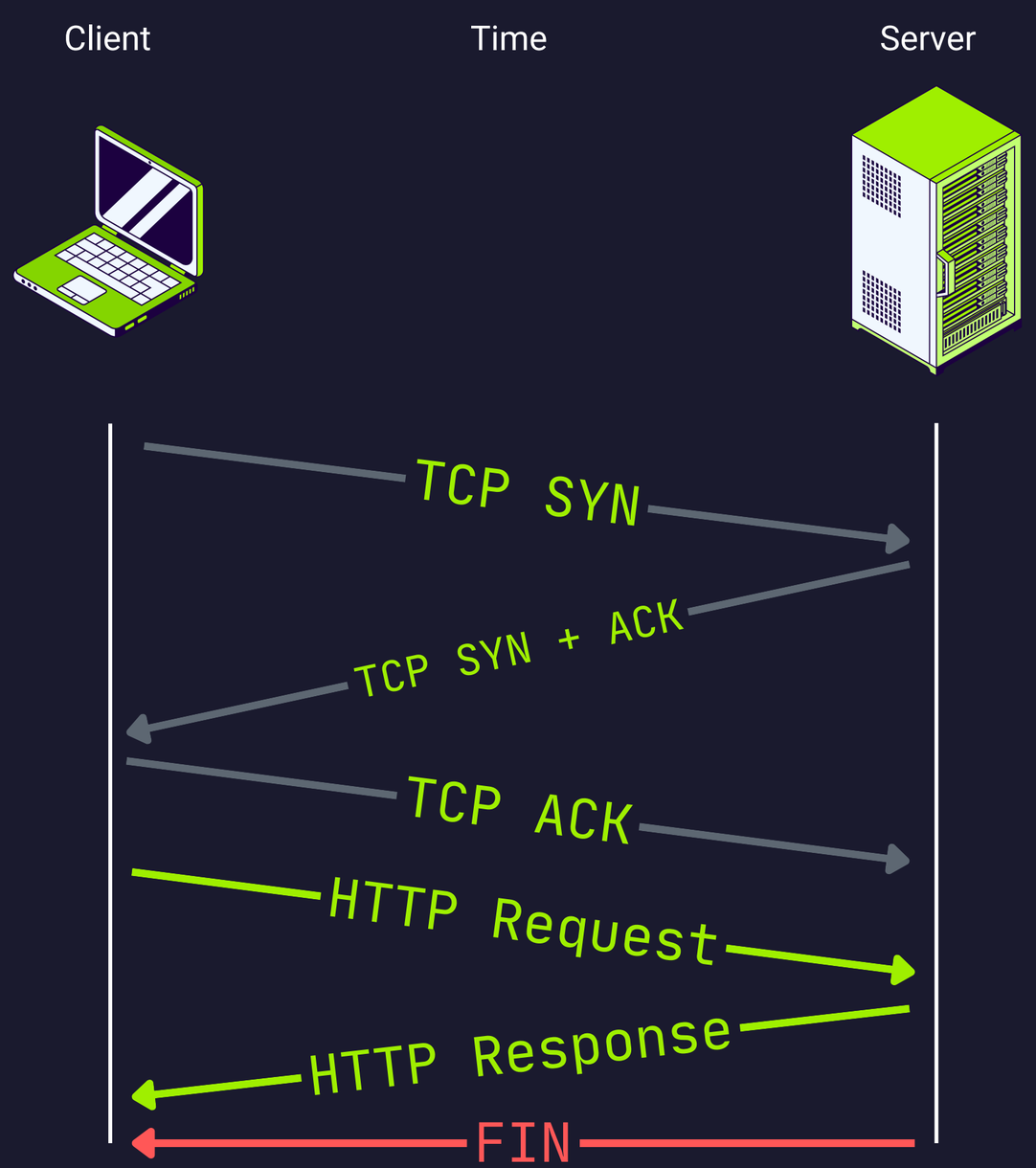
Ph.D Mohammed Sahraoui

HTTP 1.0



Ph.D Mohammed Sahraoui

HTTP 1.0



Ph.D Mohammed Sahraoui

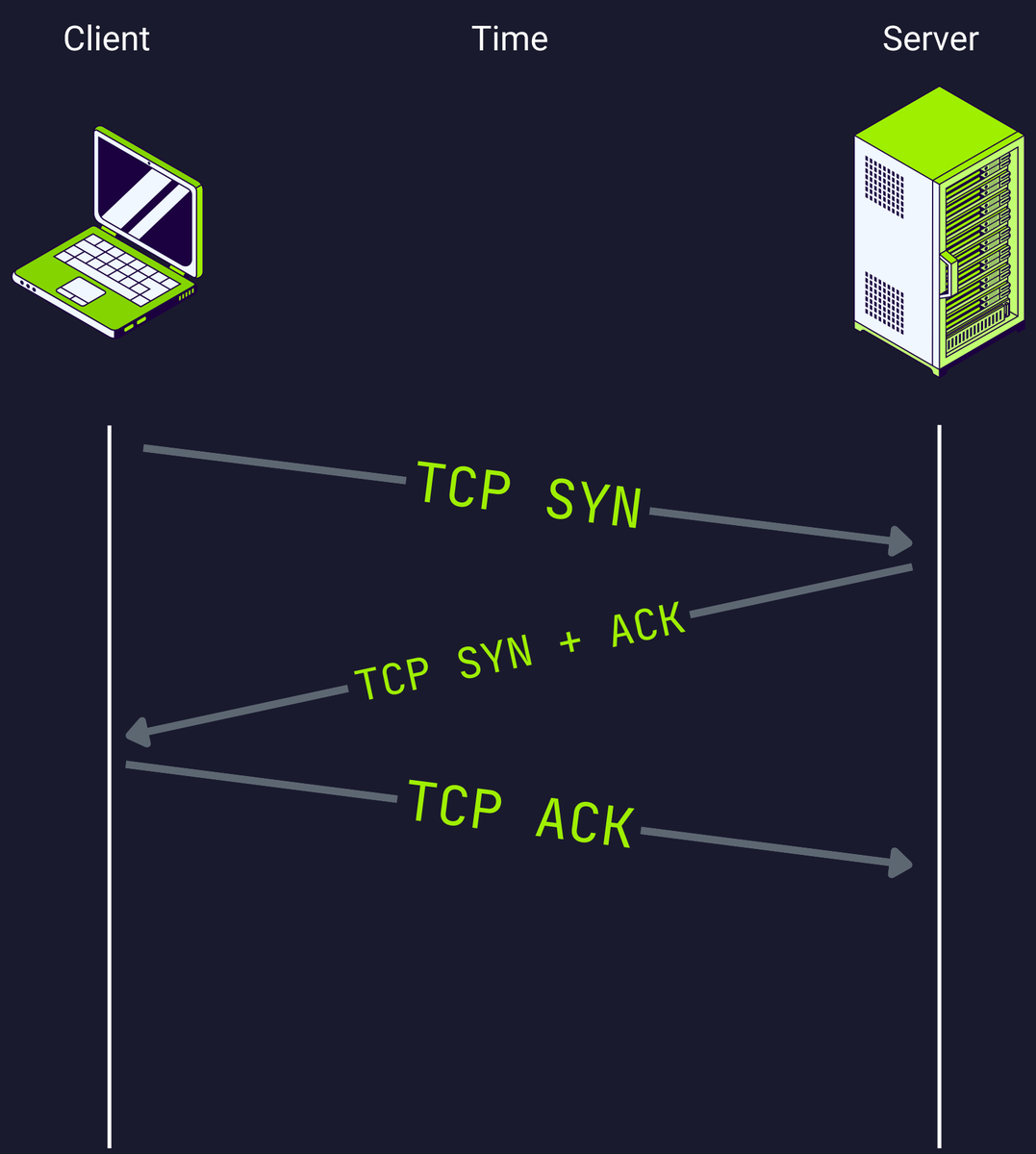
HTTP 1.1

keep-alive header

Reuse the same TCP connection for multiple HTTP requests/response

`Connection: keep-alive`

Is used in both req and response



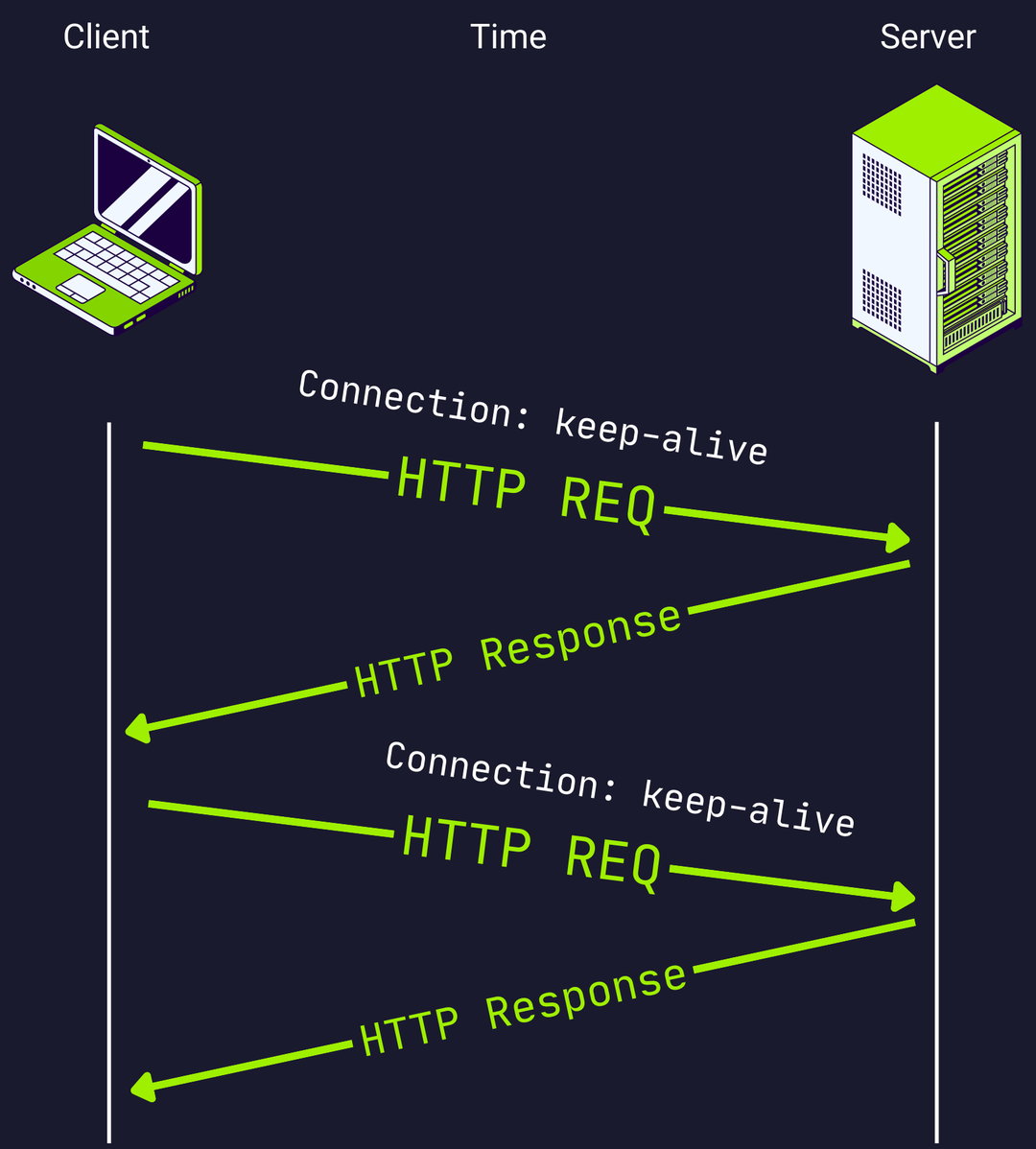
HTTP 1.1

keep-alive header

Reuse the same TCP connection for multiple HTTP requests/response

`Connection: keep-alive`

Is used in both req and response



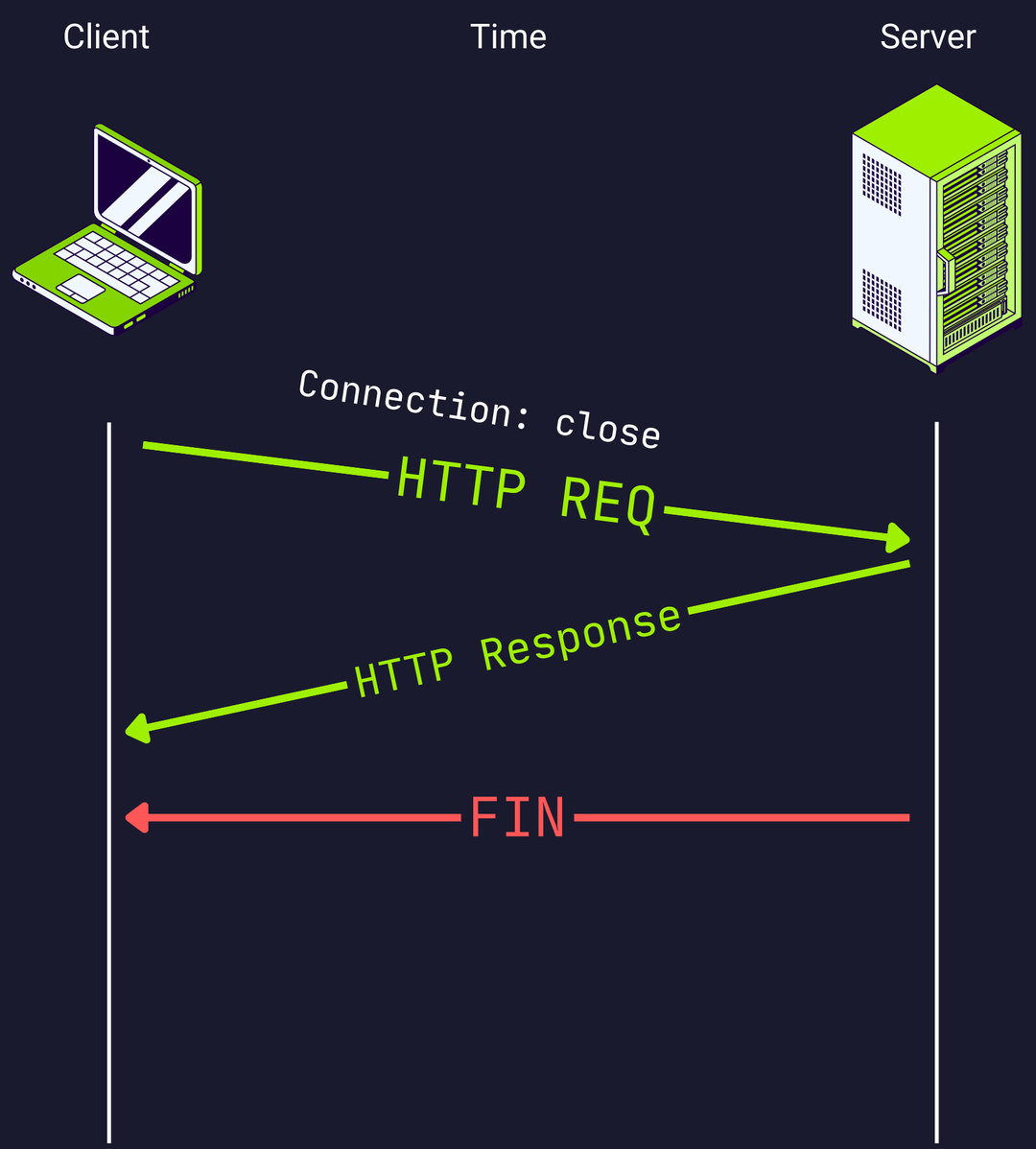
HTTP 1.1

keep-alive header

Reuse the same TCP connection for multiple HTTP requests/response

`Connection: keep-alive`

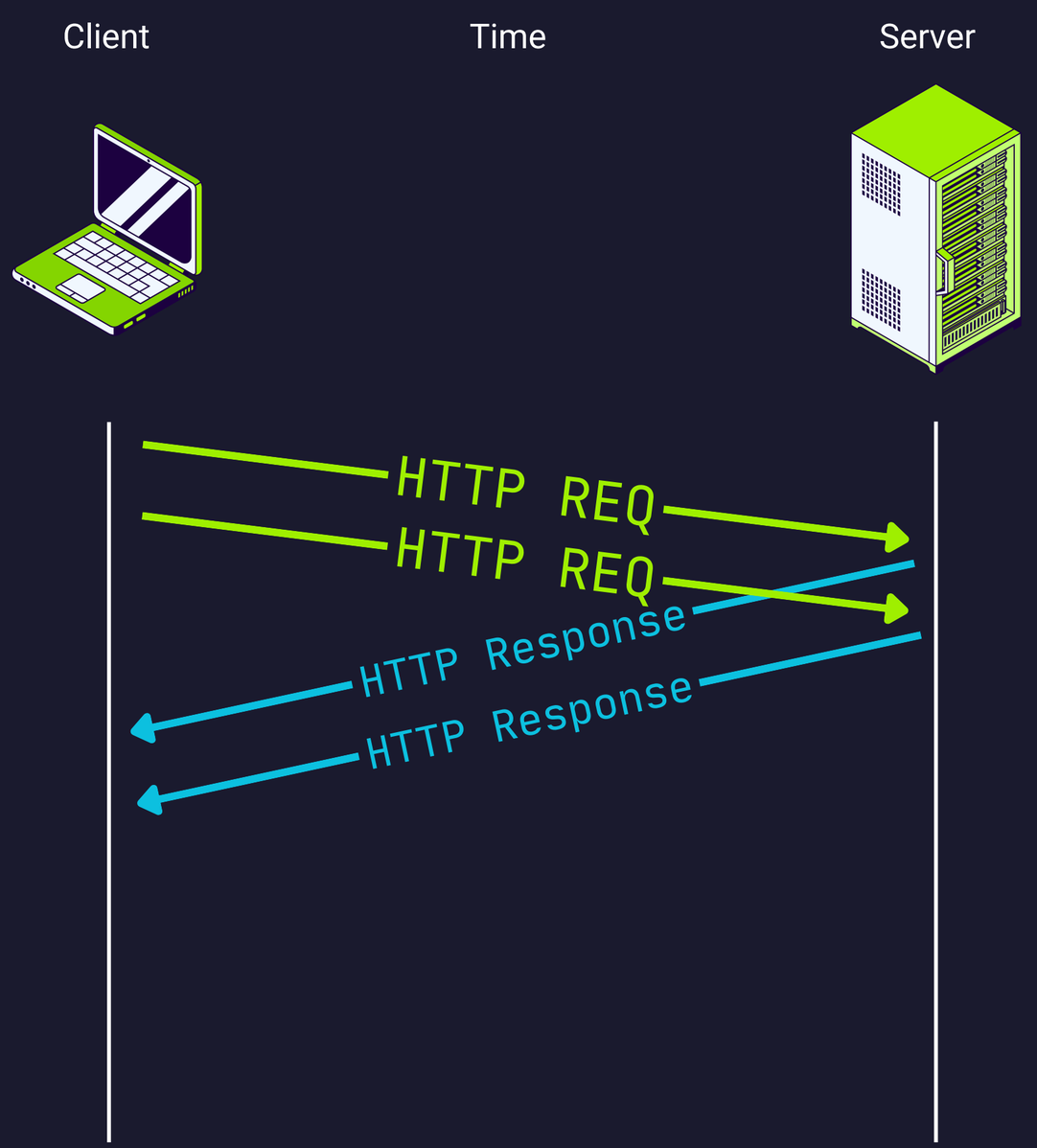
Is used in both req and response



HTTP 1.1

pipelining

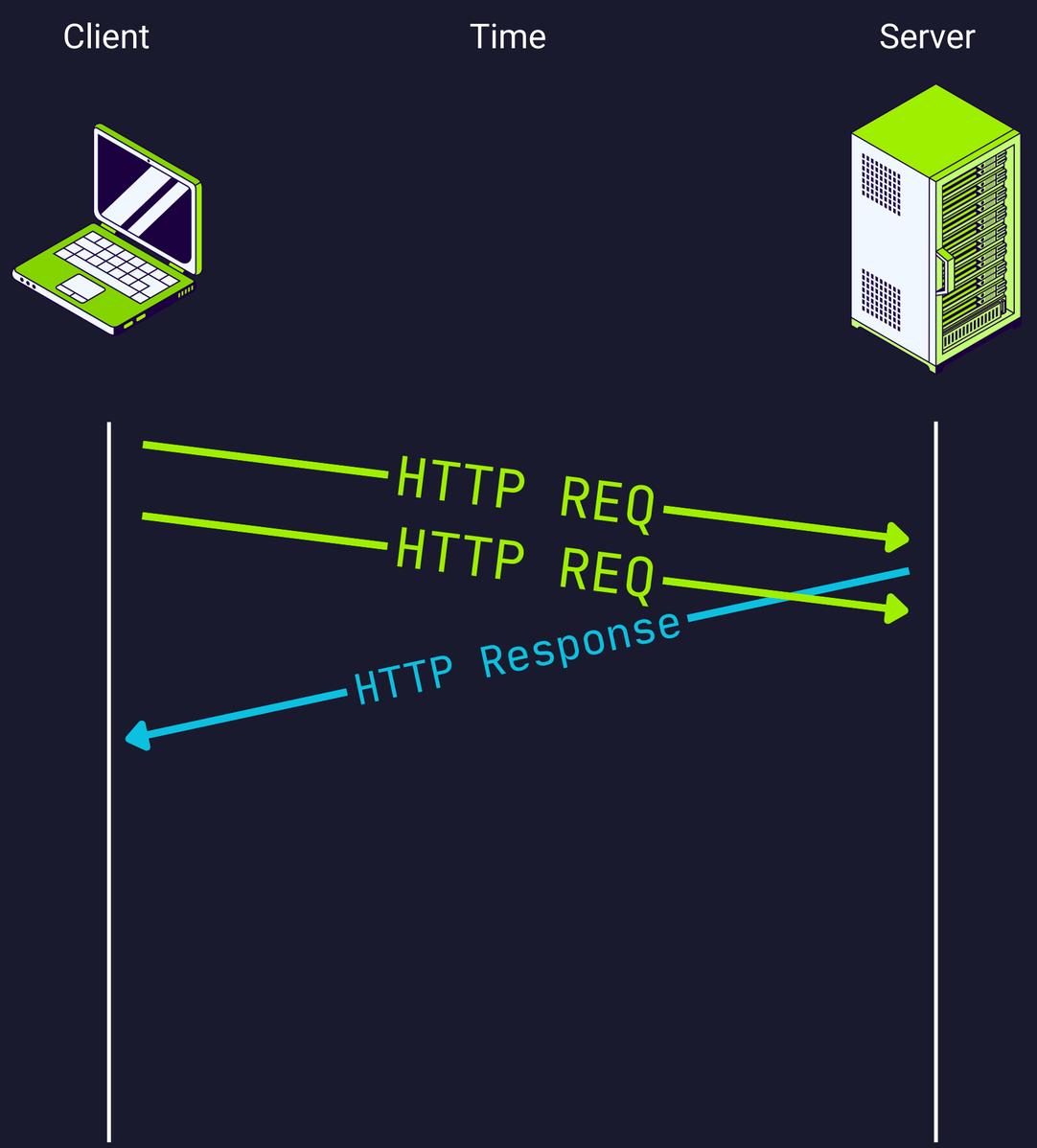
Sending multiple request at once, then receive multiple responses



HTTP 1.1

head-of-line blocking

Subsequent requests must wait for the completion of the previous request.

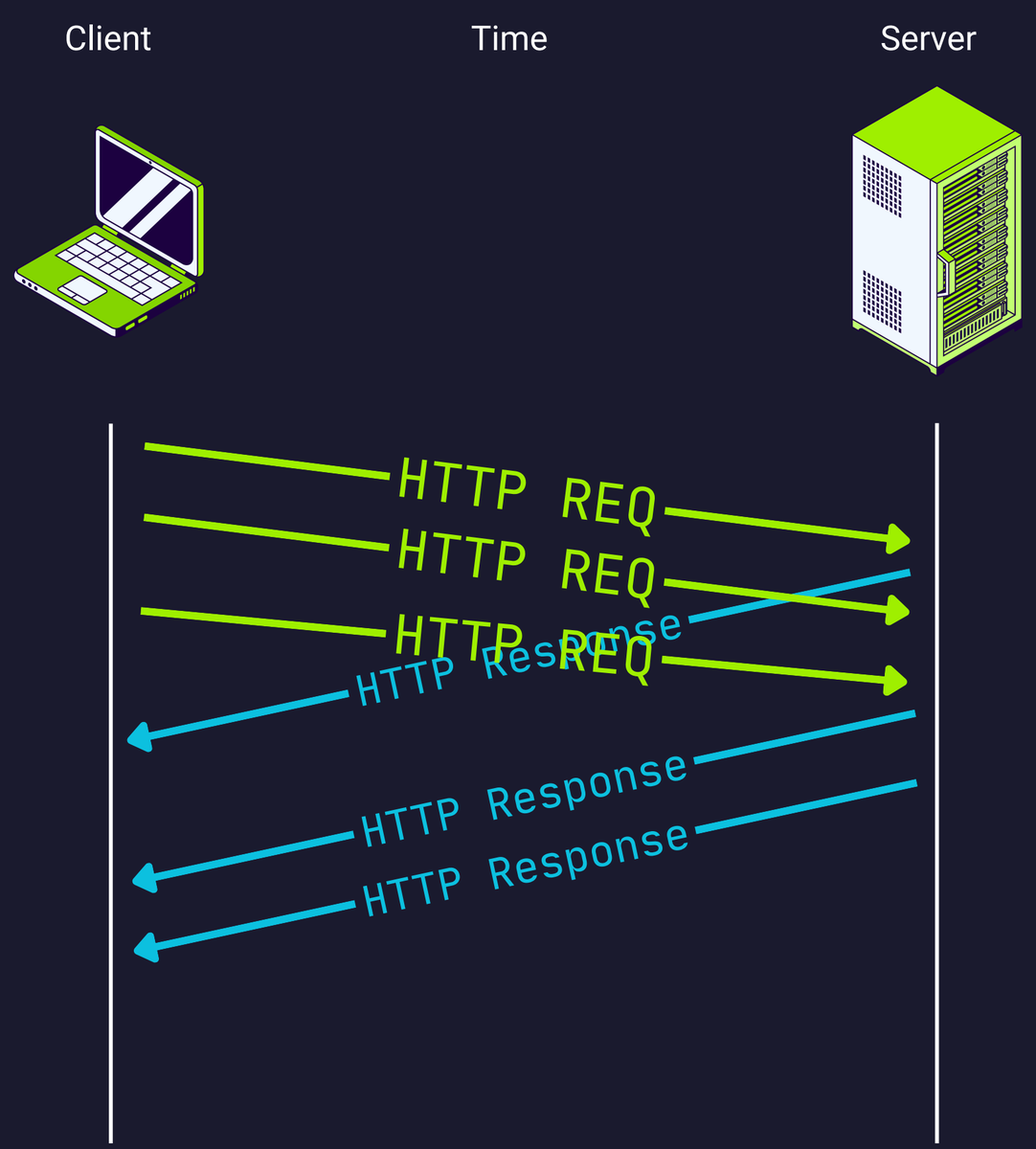


HTTP 1.1

head-of-line blocking

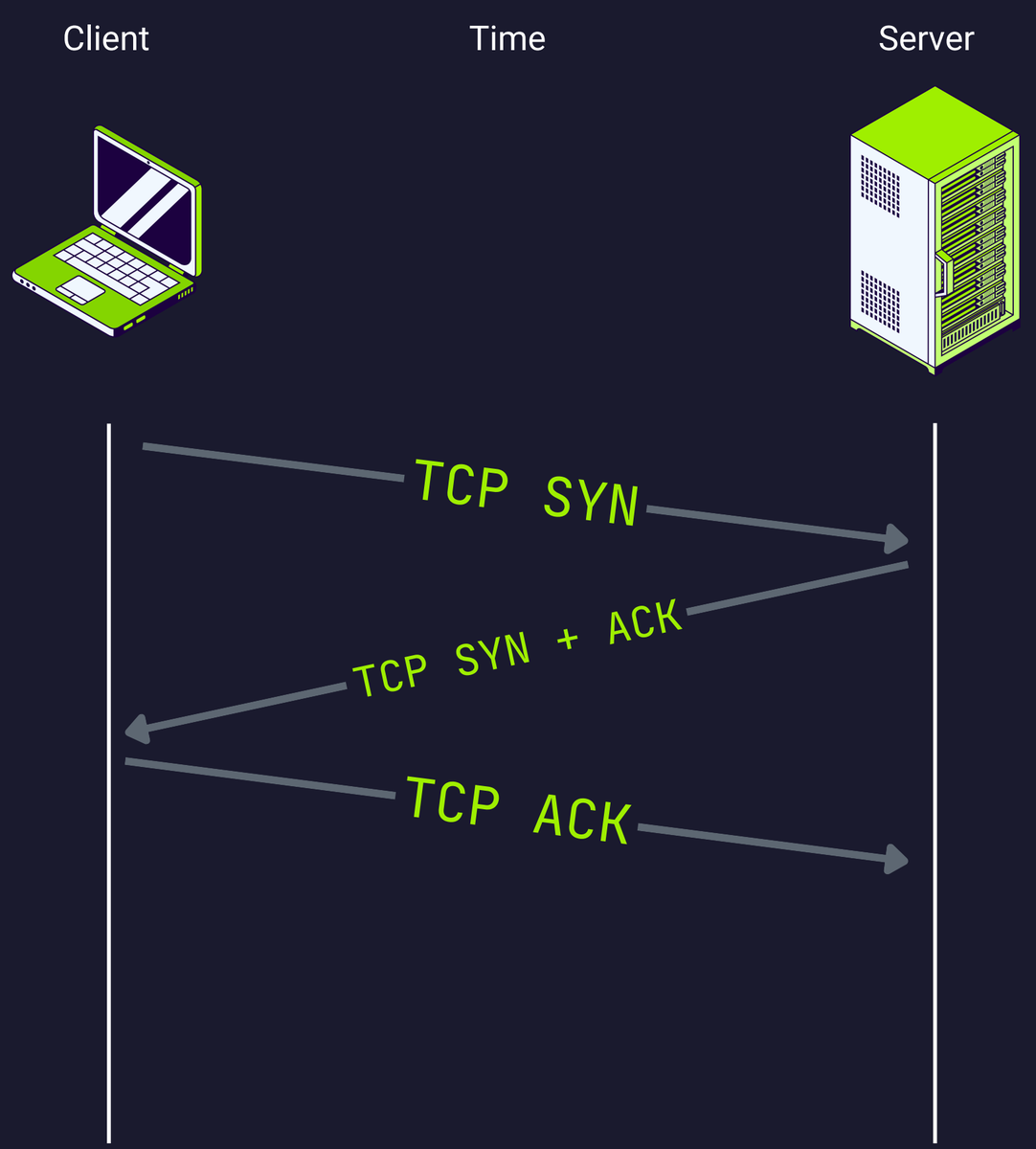
Subsequent requests must wait for the completion of the previous request.

eg. packet lost



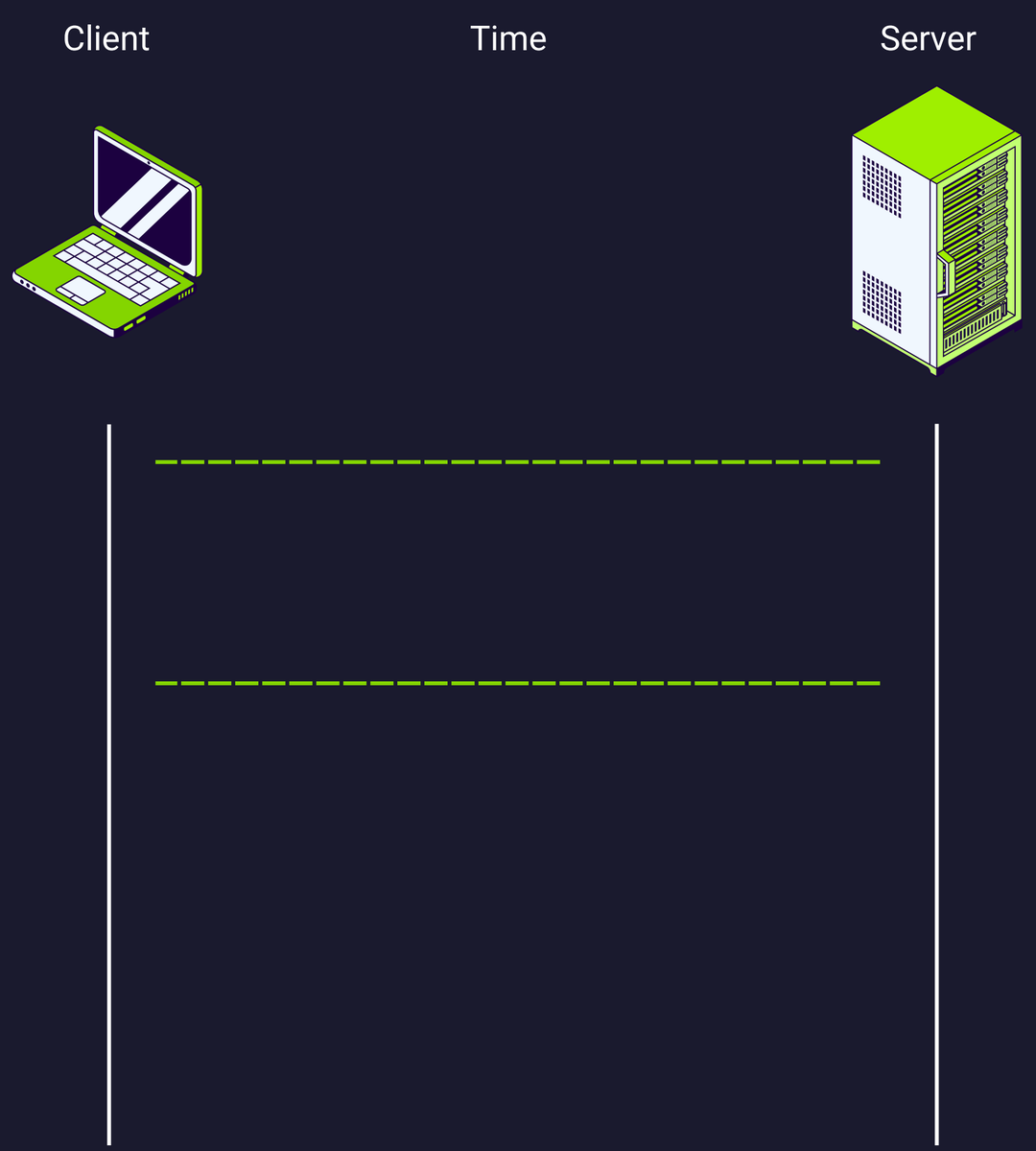
HTTP 2

HTTP streams and compressed headers



HTTP 2

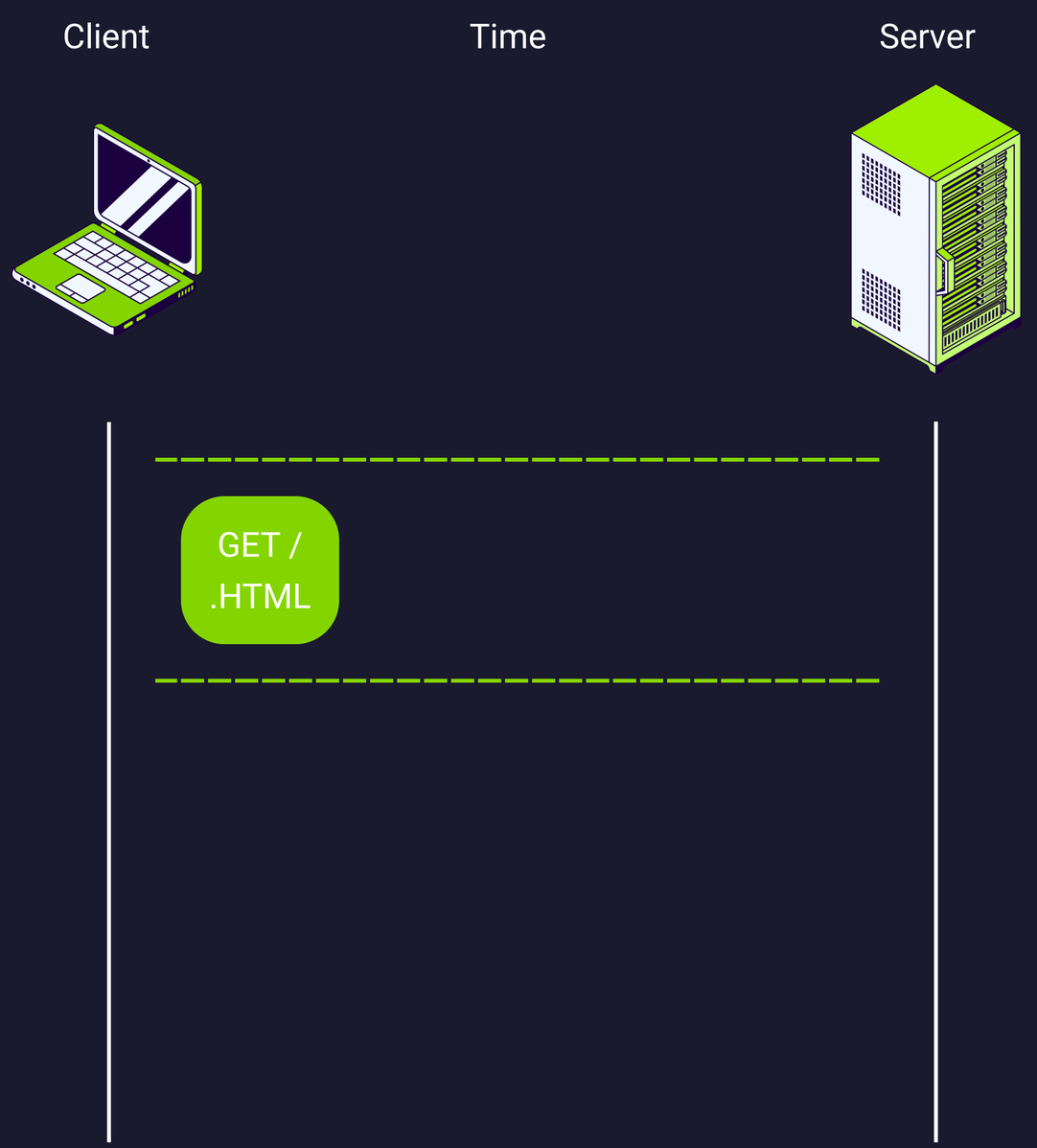
HTTP streams and compressed headers



Ph.D Mohammed Sahraoui

HTTP 2

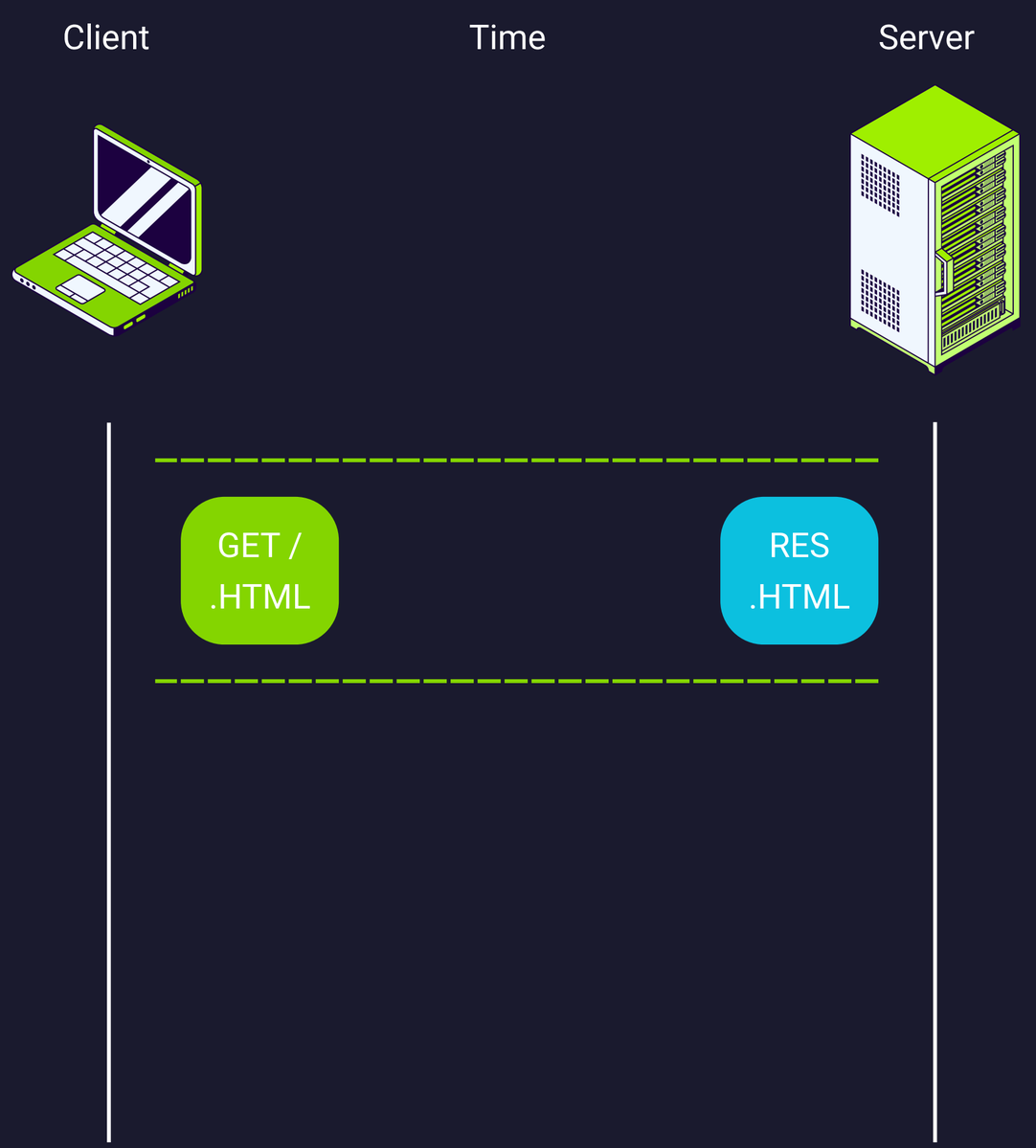
HTTP streams and compressed headers



Ph.D Mohammed Sahraoui

HTTP 2

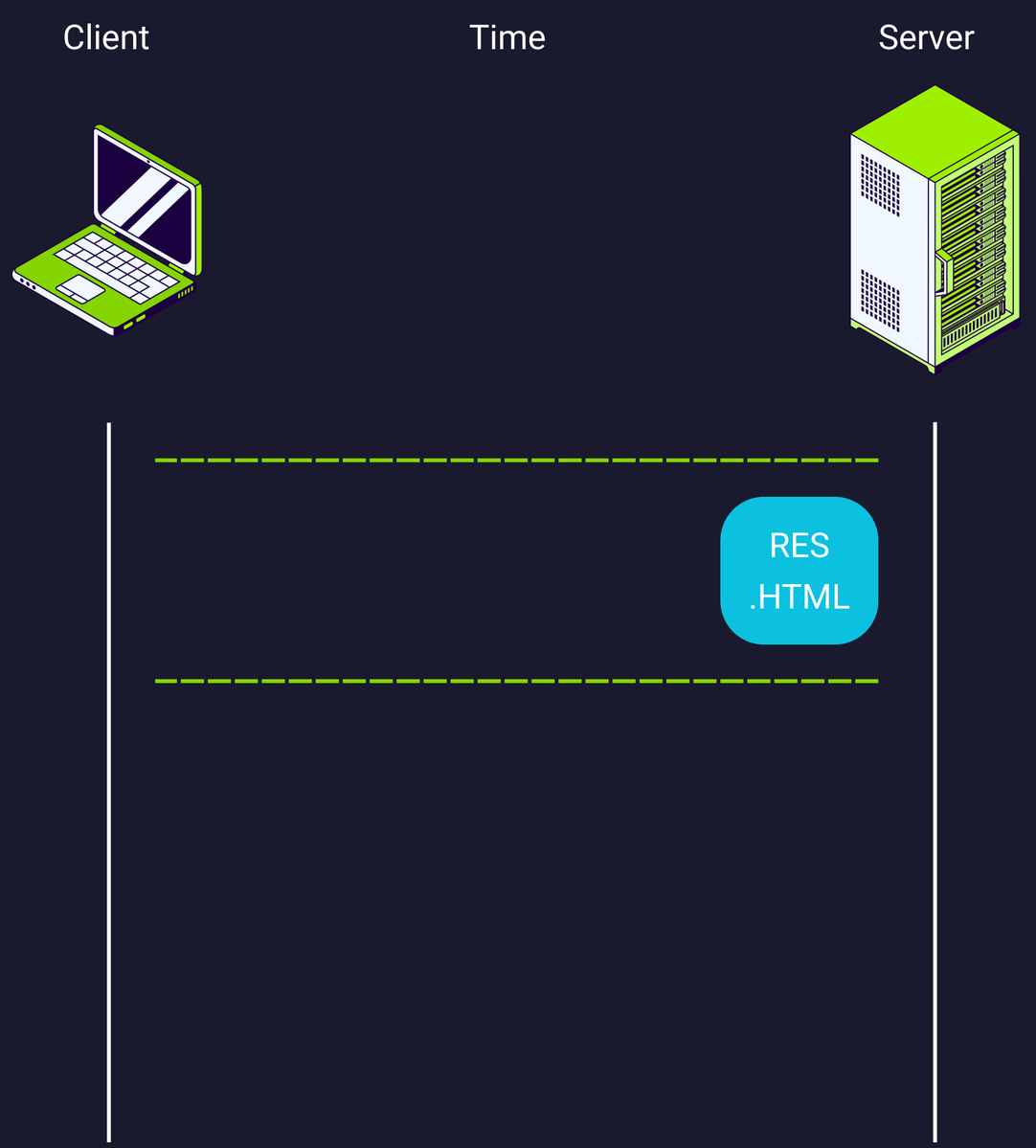
HTTP streams and compressed headers



Ph.D Mohammed Sahraoui

HTTP 2

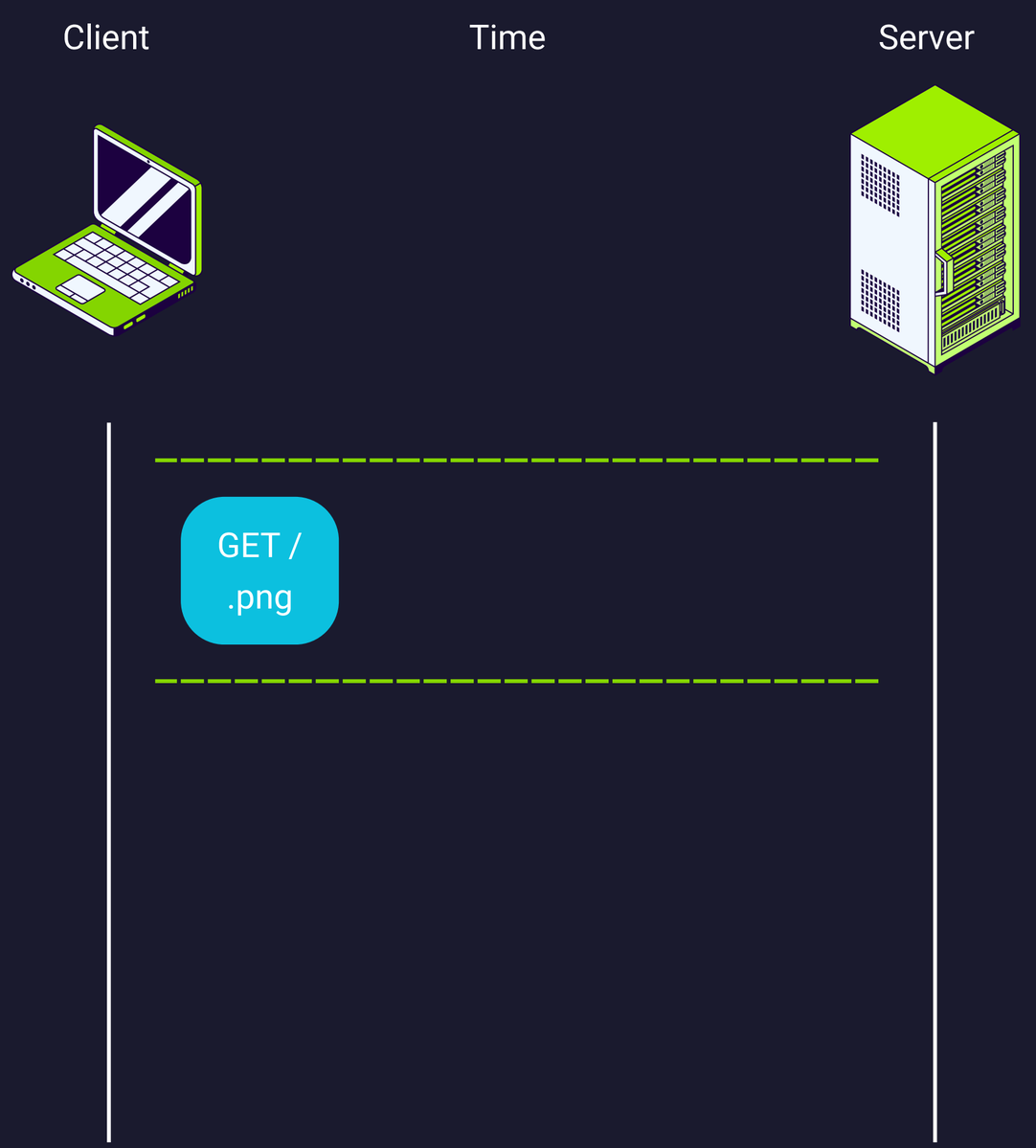
HTTP streams and compressed headers



Ph.D Mohammed Sahraoui

HTTP 2

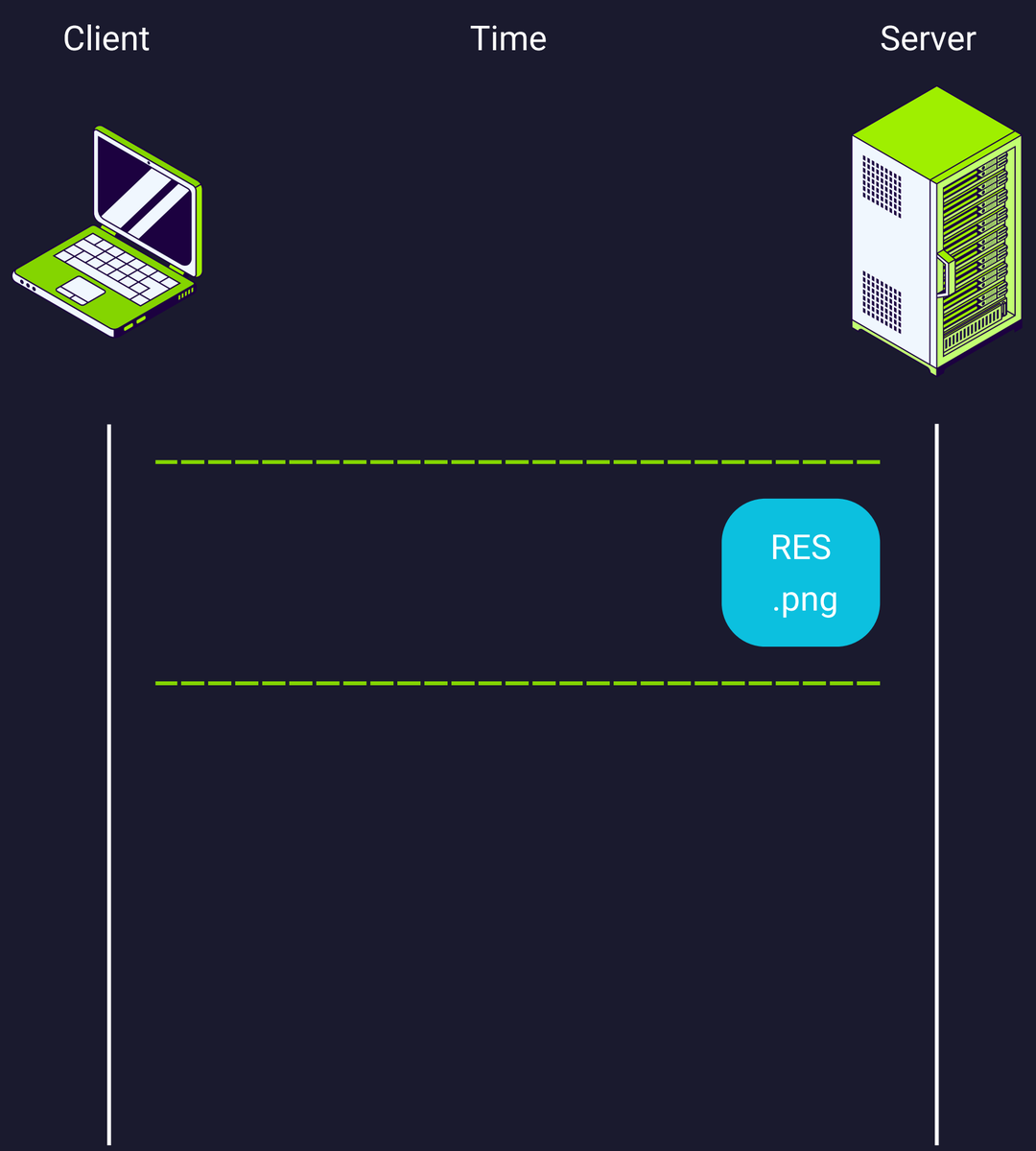
HTTP streams and compressed headers



Ph.D Mohammed Sahraoui

HTTP 2

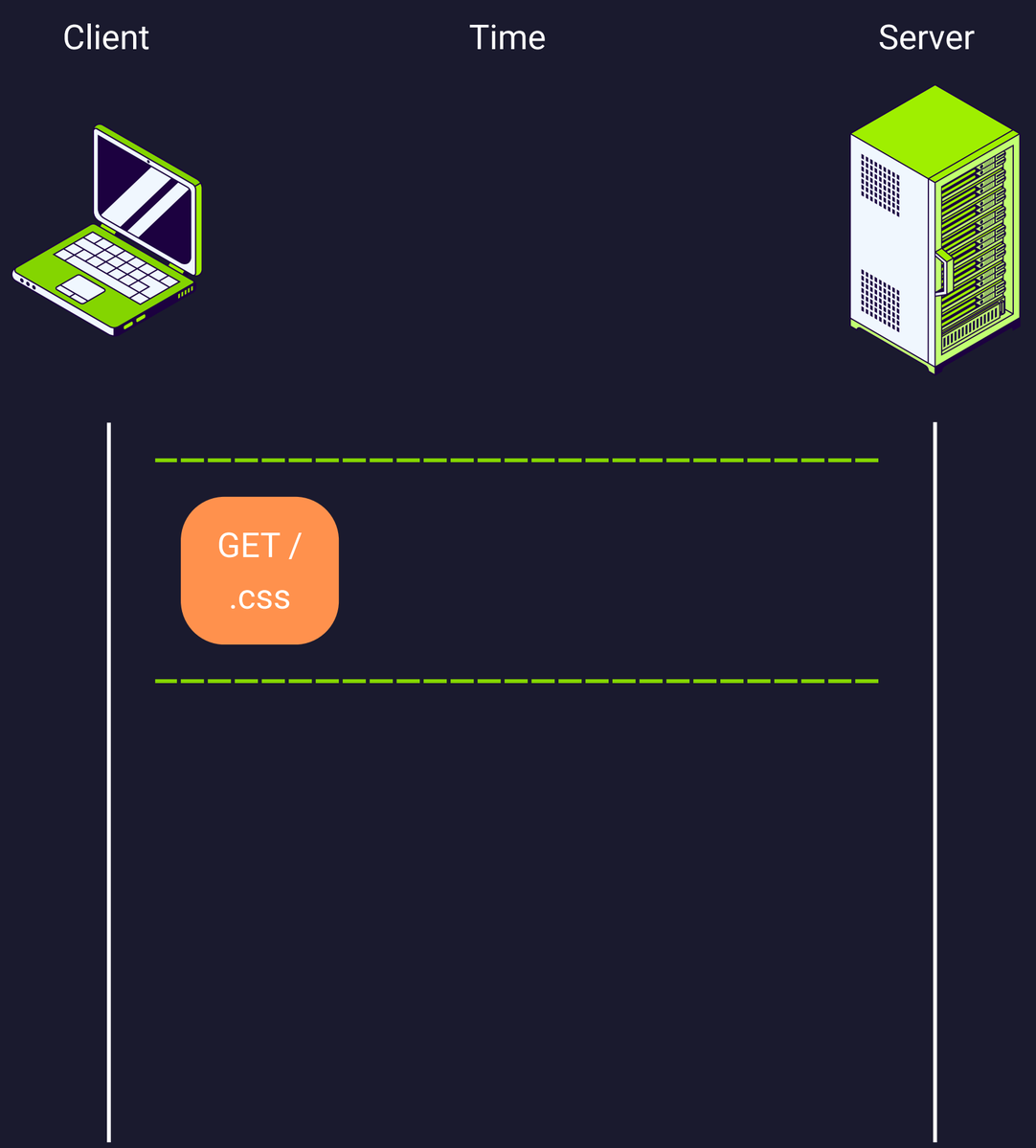
HTTP streams and compressed headers



Ph.D Mohammed Sahraoui

HTTP 2

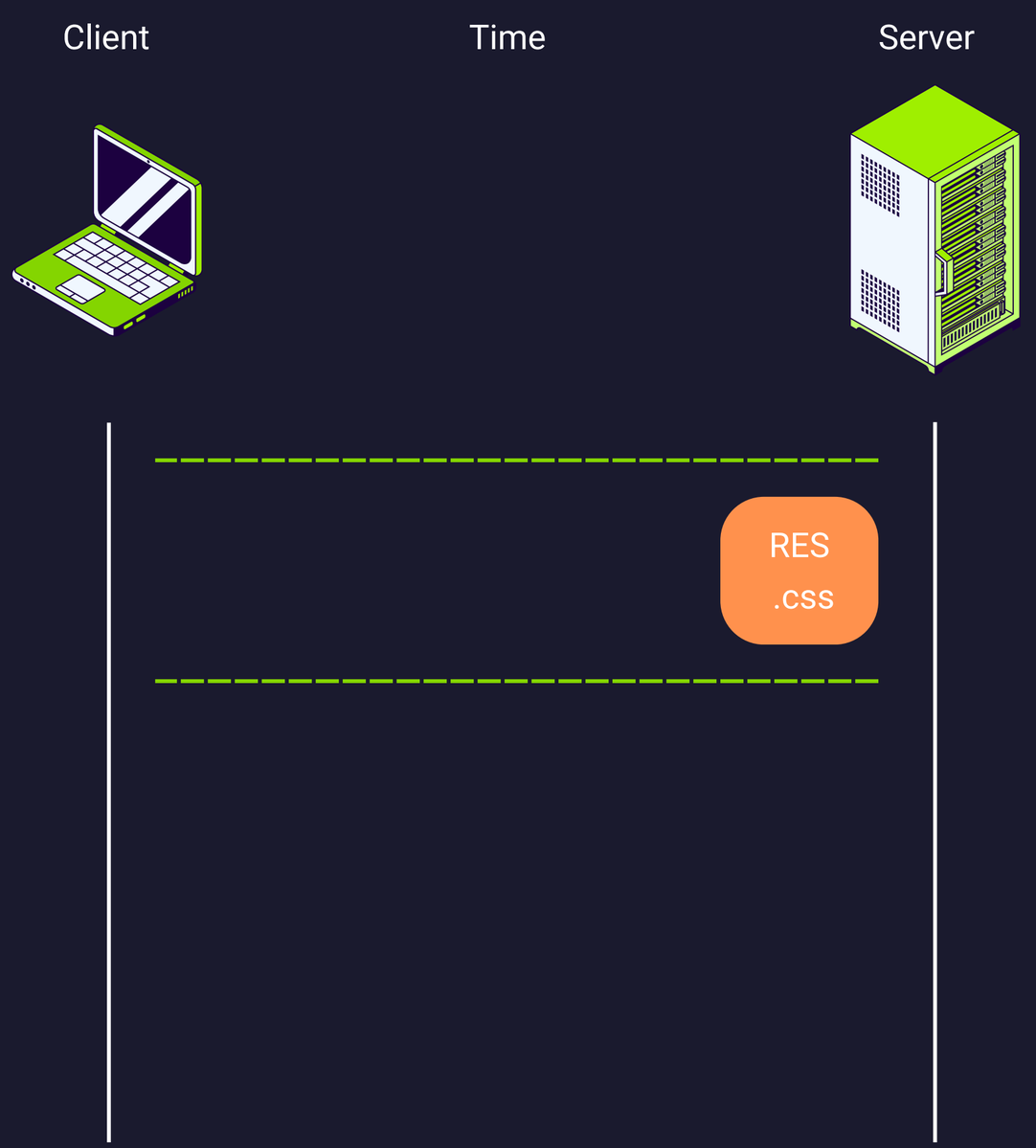
HTTP streams and compressed headers



Ph.D Mohammed Sahraoui

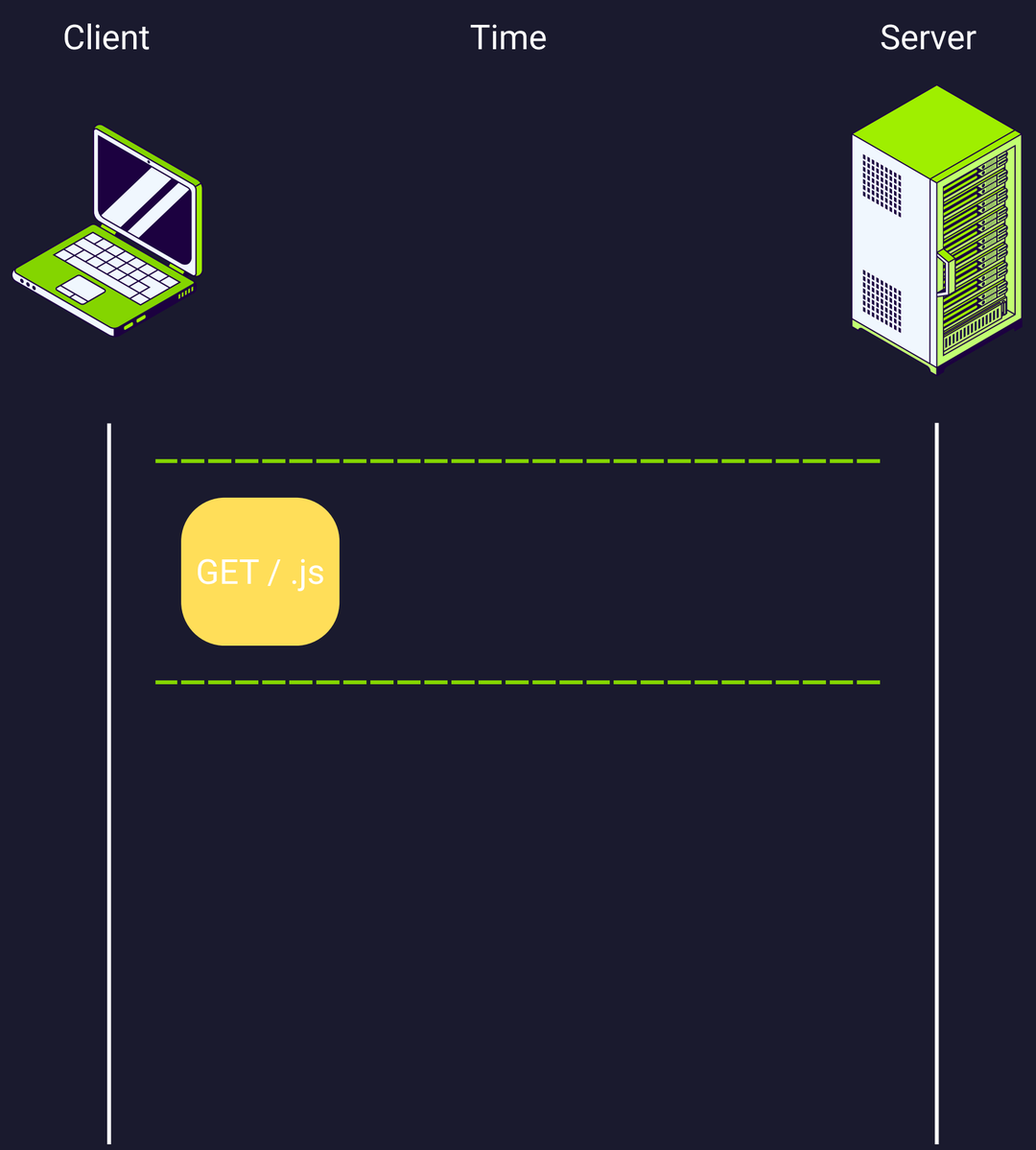
HTTP 2

HTTP streams and compressed headers



HTTP 2

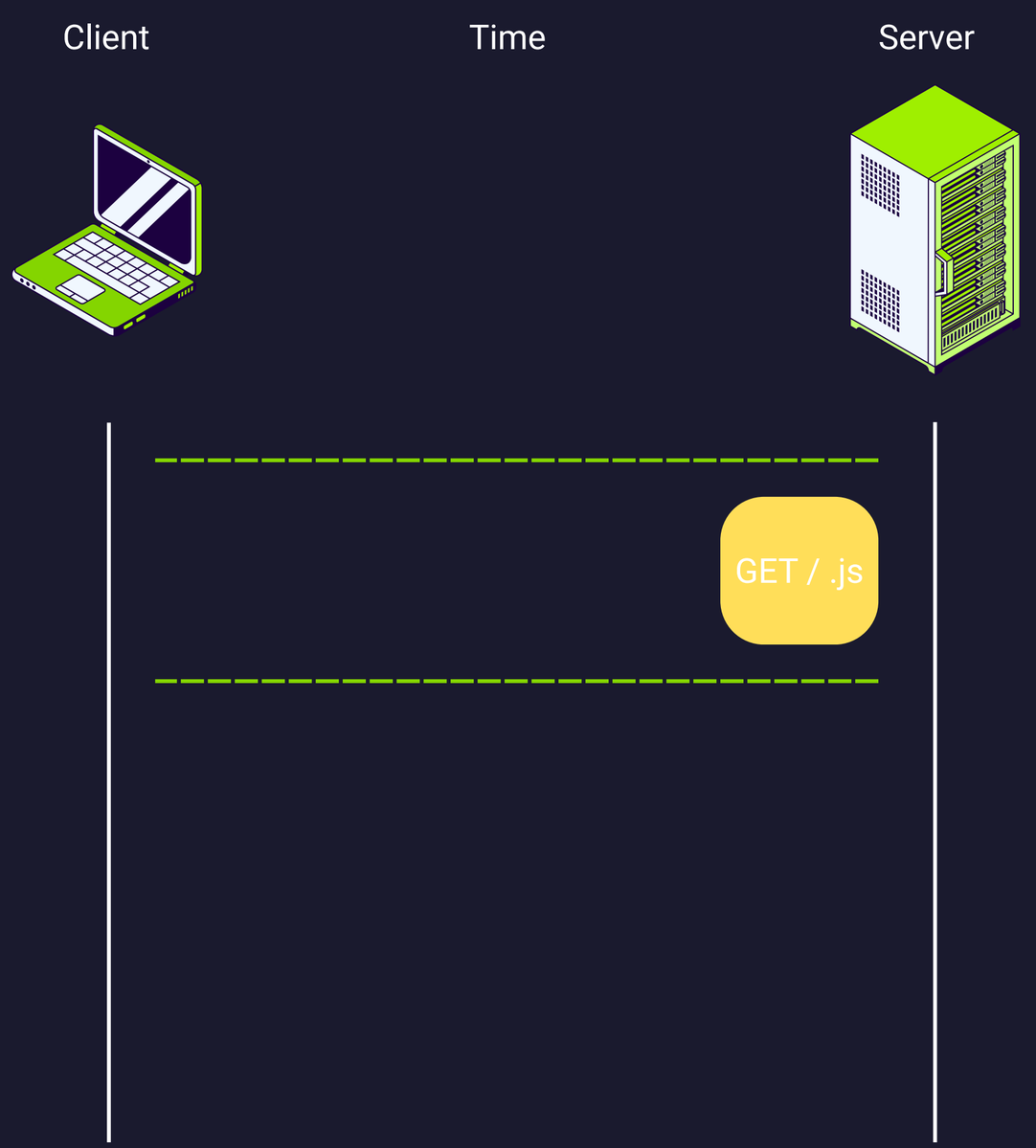
HTTP streams and compressed headers



Ph.D Mohammed Sahraoui

HTTP 2

HTTP streams and compressed headers



Ph.D Mohammed Sahraoui

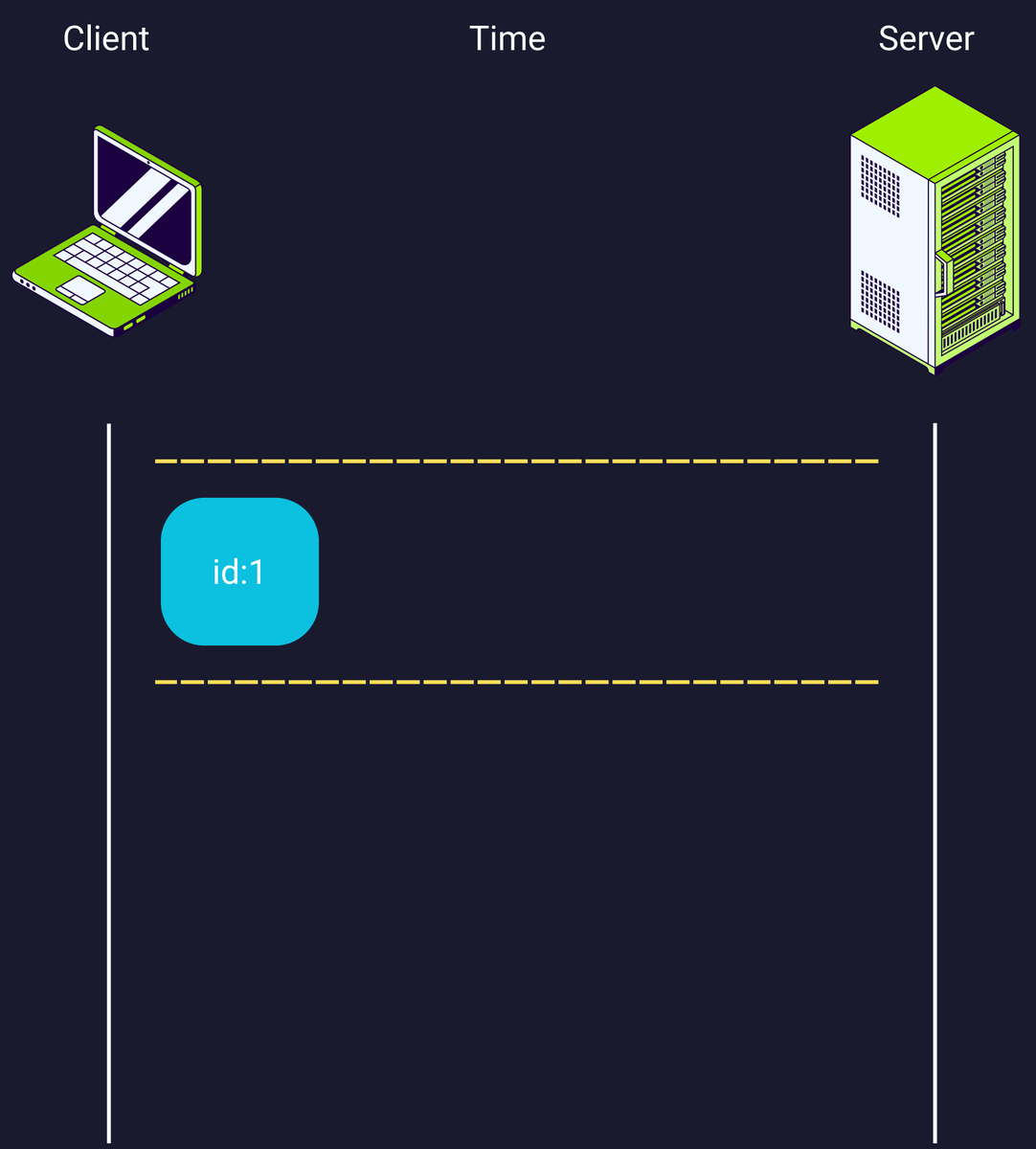
HTTP 3

HTTP streams as first-class citizens.

Based on UDP (QUIC)

Provide much speed

Provide security standards



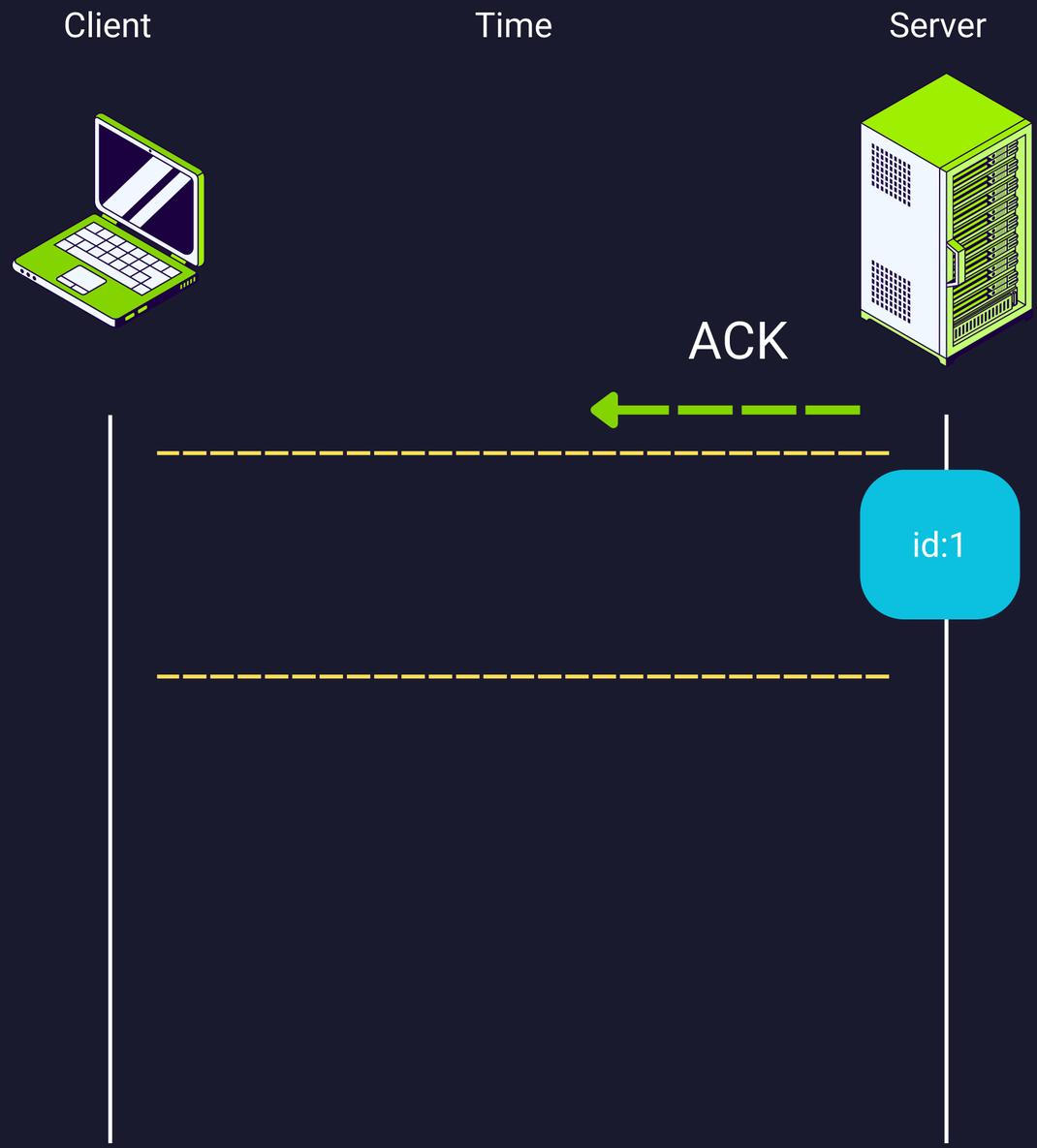
HTTP 3

HTTP streams as first-class citizens.

Based on UDP (QUIC)

Provide much speed

Provide security standards



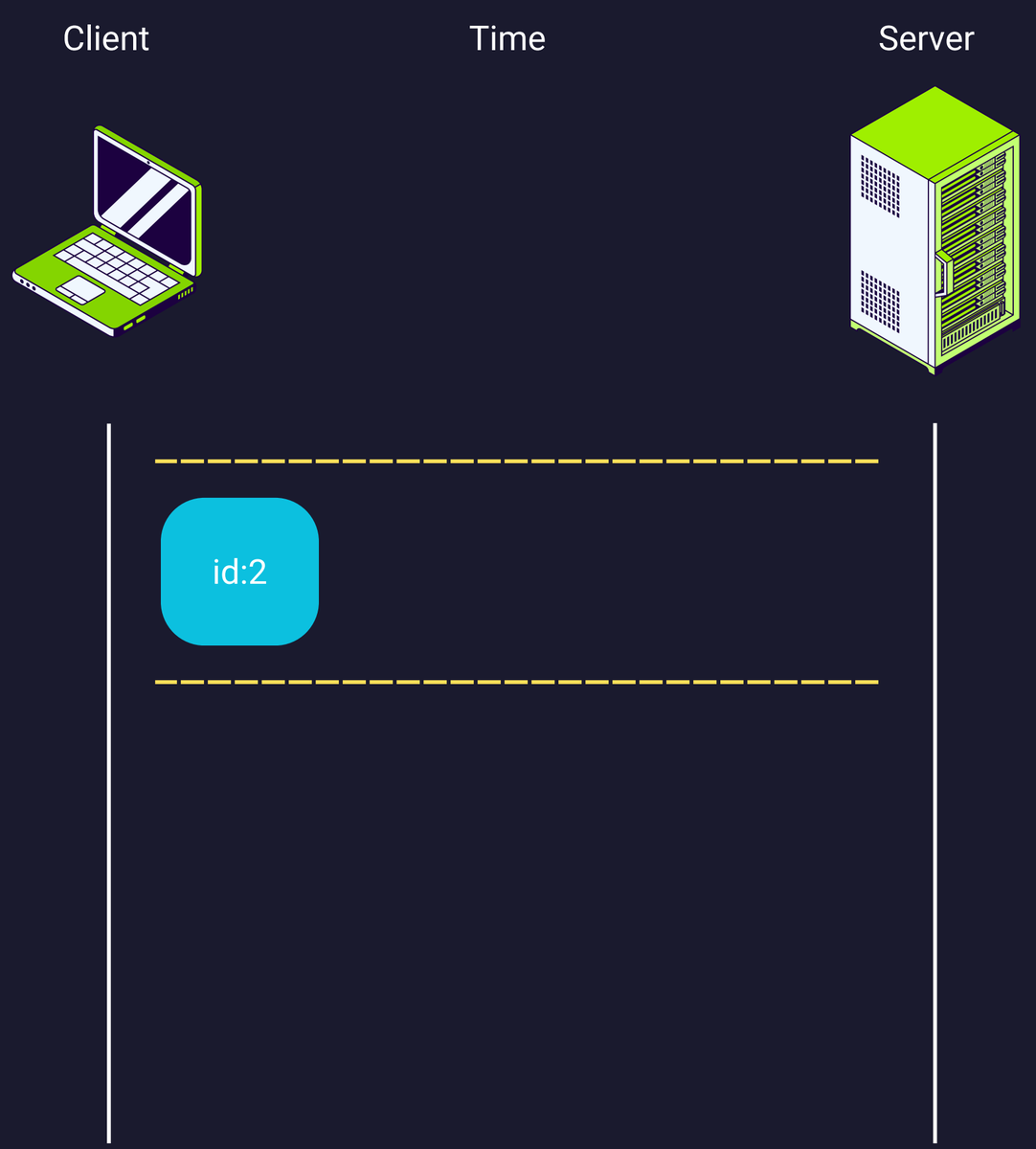
HTTP 3

HTTP streams as first-class citizens.

Based on UDP (QUIC)

Provide much speed

Provide security standards



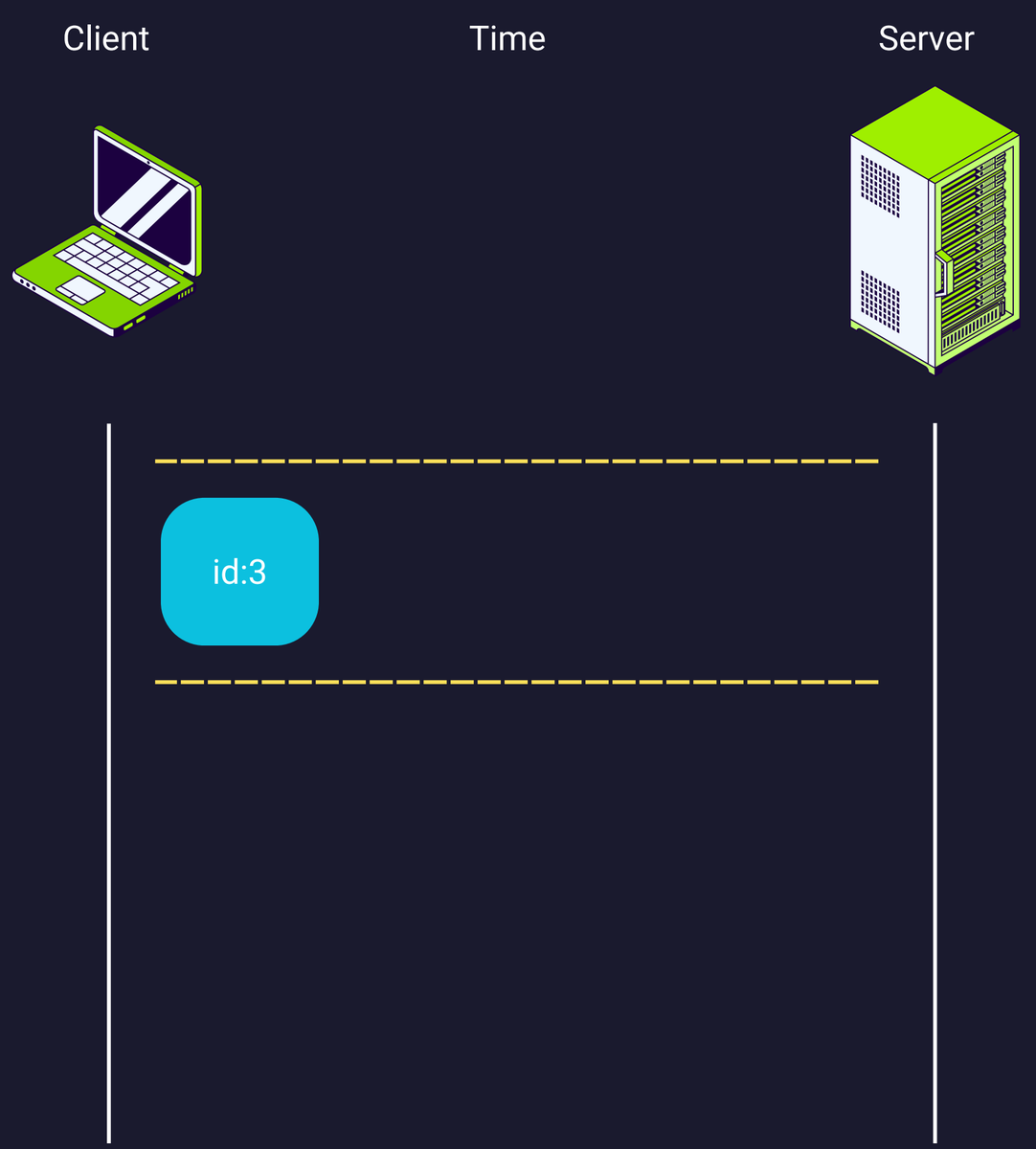
HTTP 3

HTTP streams as first-class citizens.

Based on UDP (QUIC)

Provide much speed

Provide security standards



Ph.D Mohammed Sahraoui

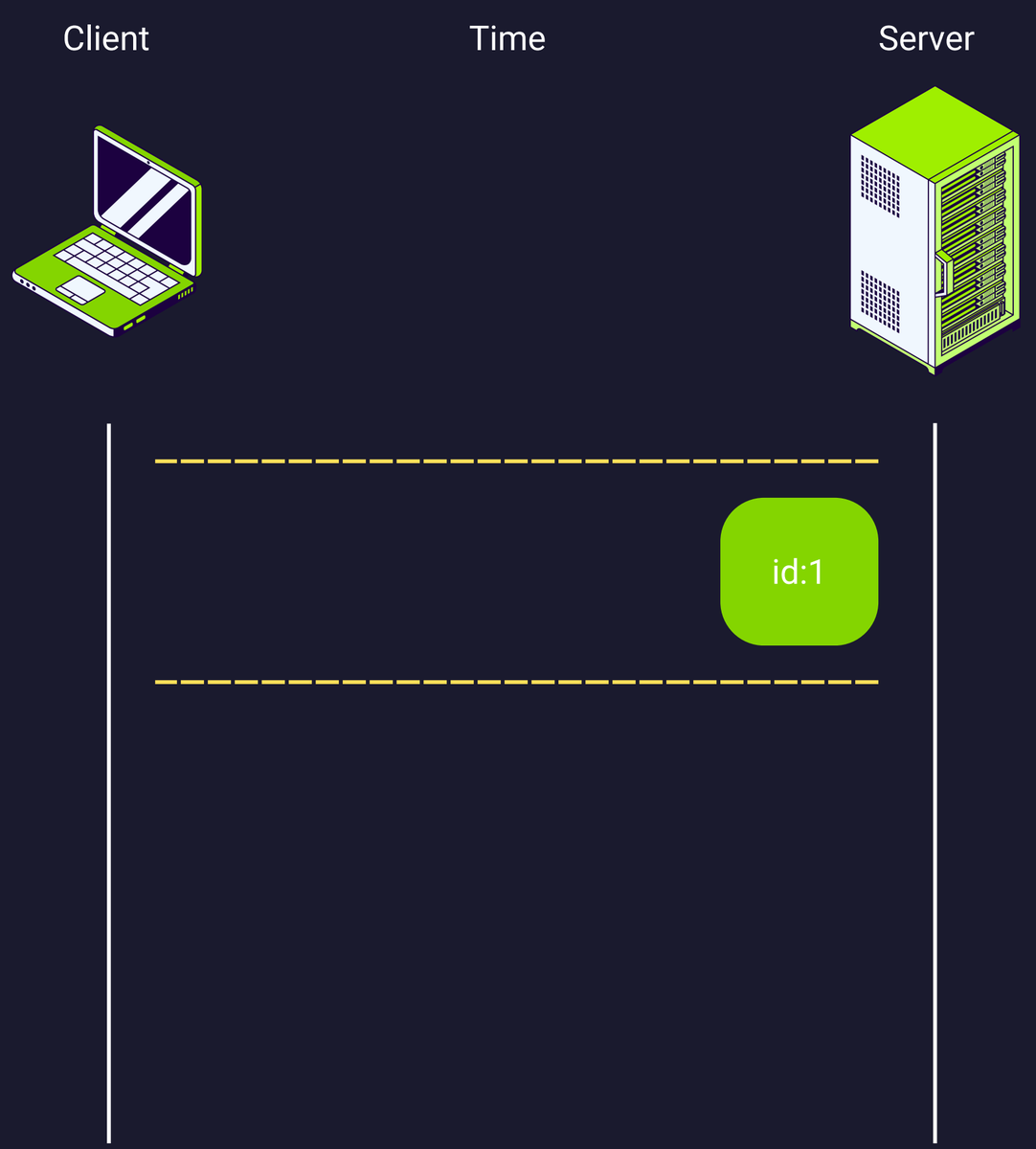
HTTP 3

HTTP streams as first-class citizens.

Based on UDP (QUIC)

Provide much speed

Provide security standards



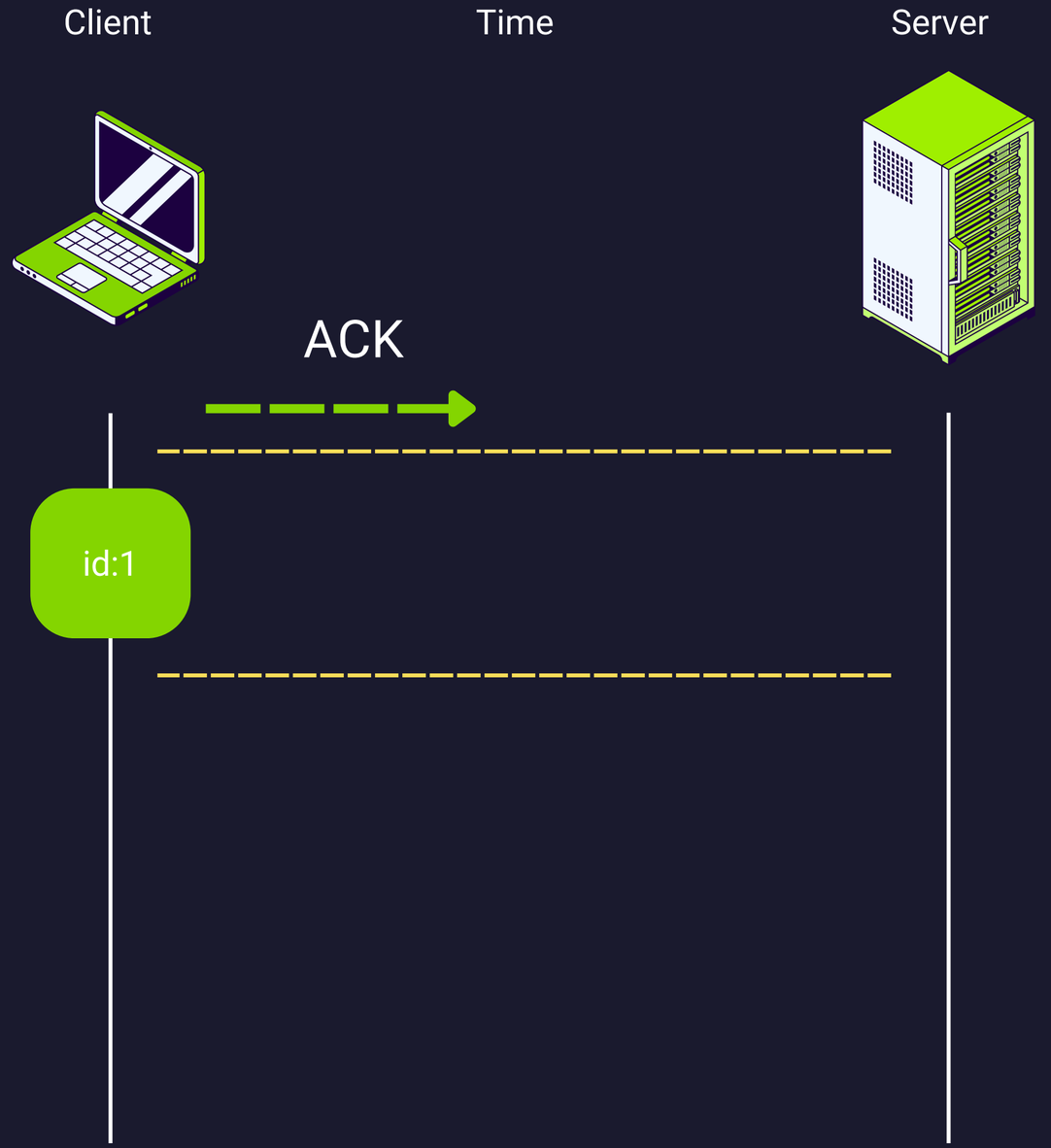
HTTP 3

HTTP streams as first-class citizens.

Based on UDP (QUIC)

Provide much speed

Provide security standards



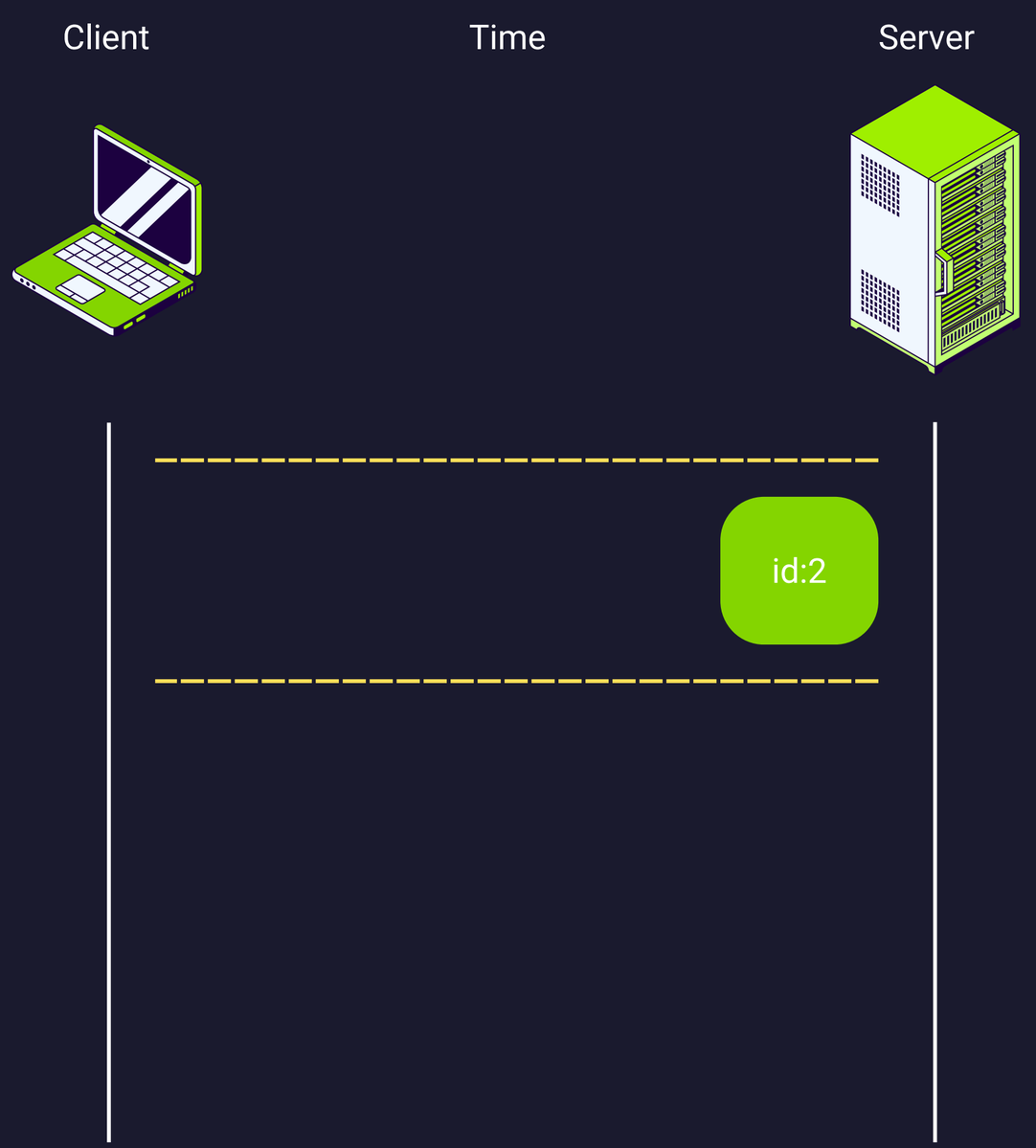
HTTP 3

HTTP streams as first-class citizens.

Based on UDP (QUIC)

Provide much speed

Provide security standards



Ph.D Mohammed Sahraoui

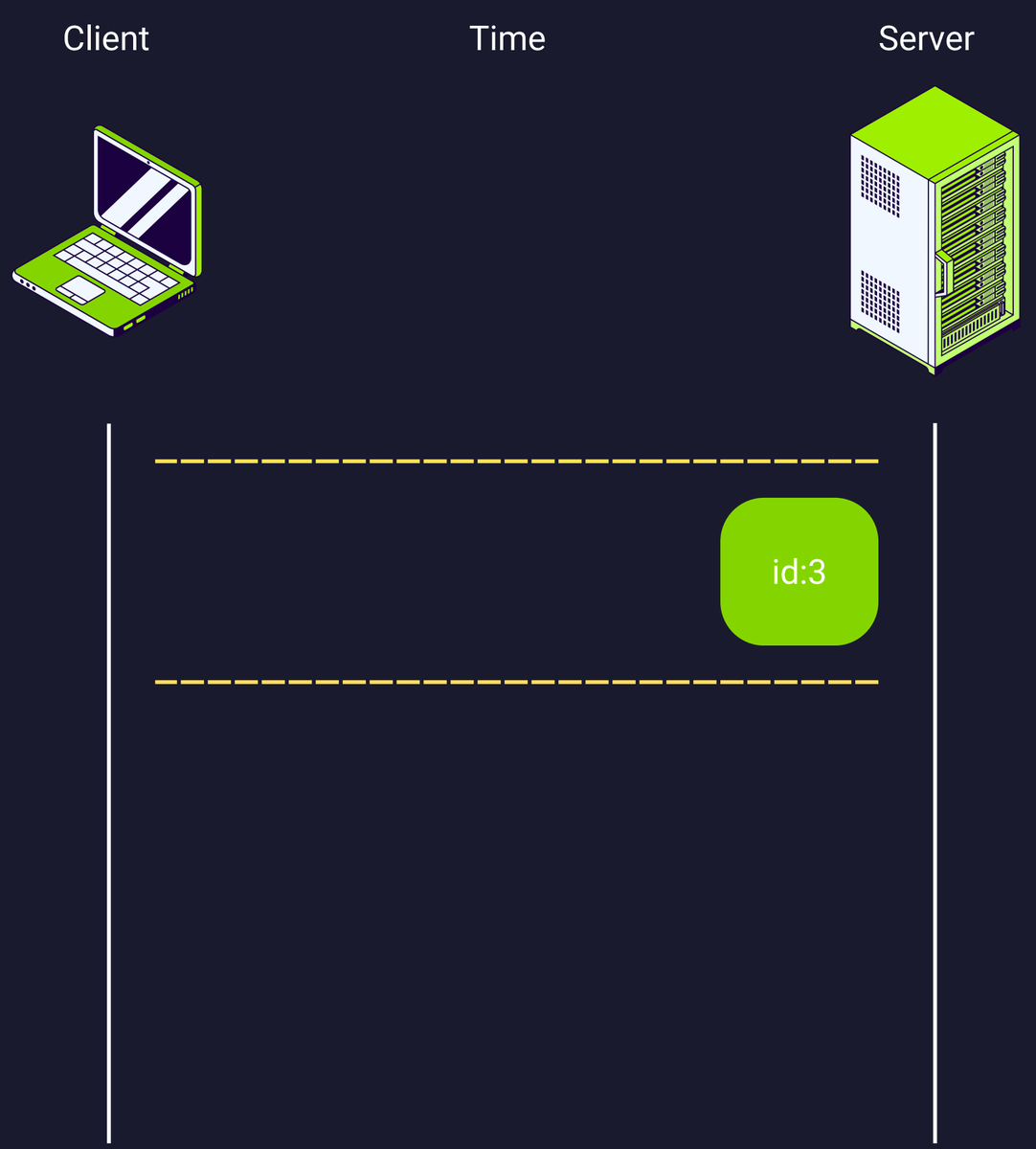
HTTP 3

HTTP streams as first-class citizens.

Based on UDP (QUIC)

Provide much speed

Provide security standards



Ph.D Mohammed Sahraoui

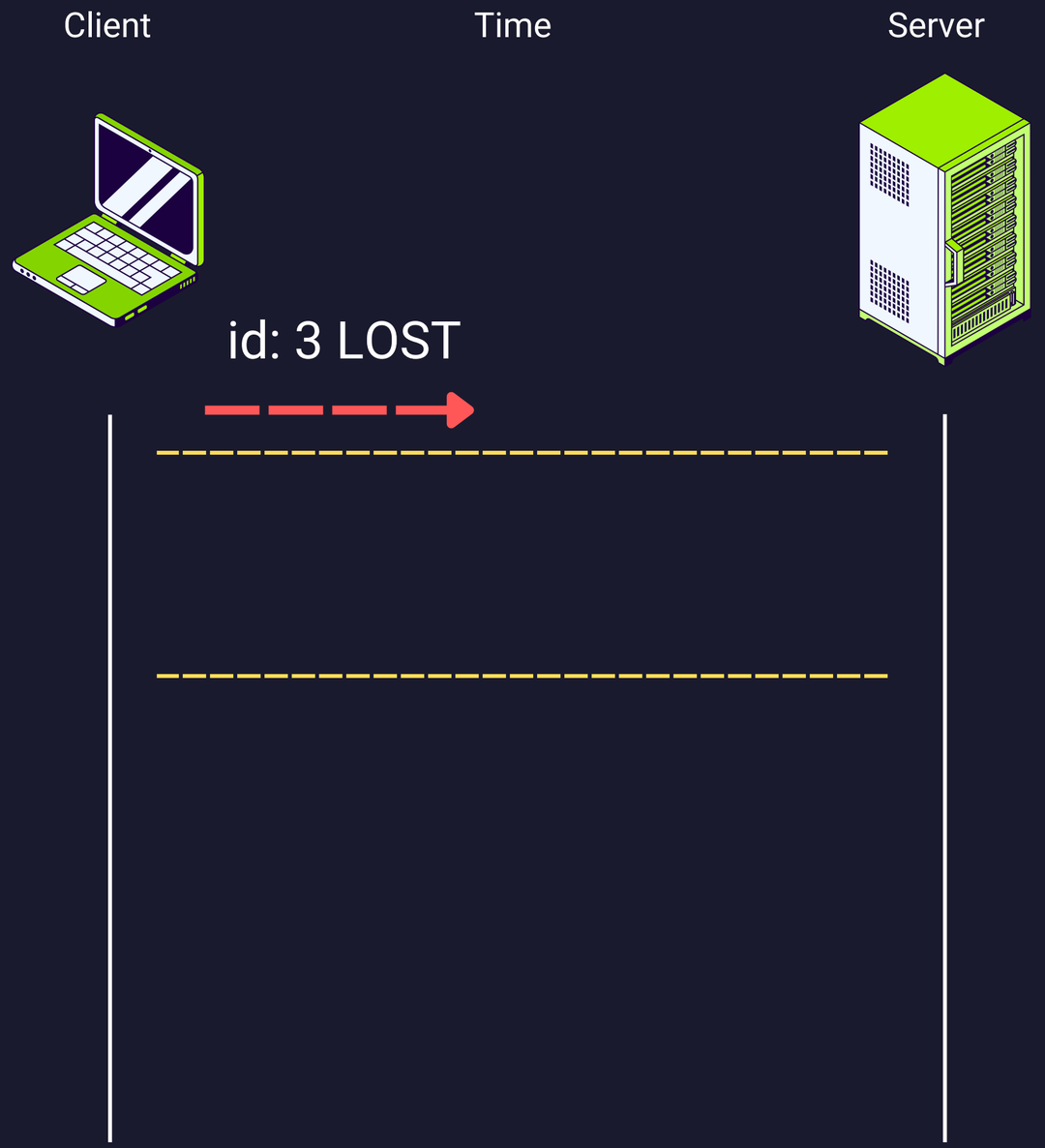
HTTP 3

HTTP streams as first-class citizens.

Based on UDP (QUIC)

Provide much speed

Provide security standards



Ph.D Mohammed Sahraoui

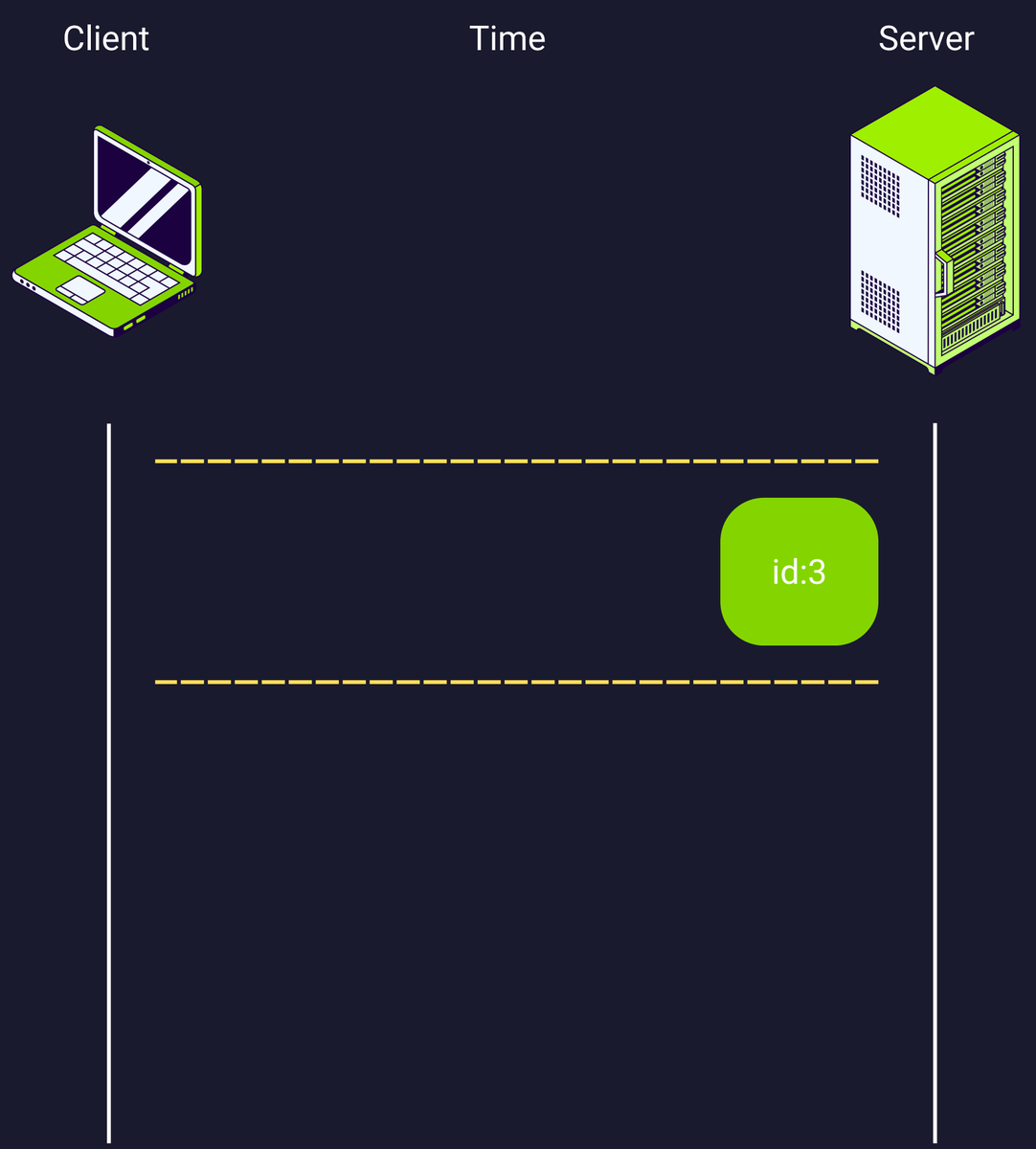
HTTP 3

HTTP streams as first-class citizens.

Based on UDP (QUIC)

Provide much speed

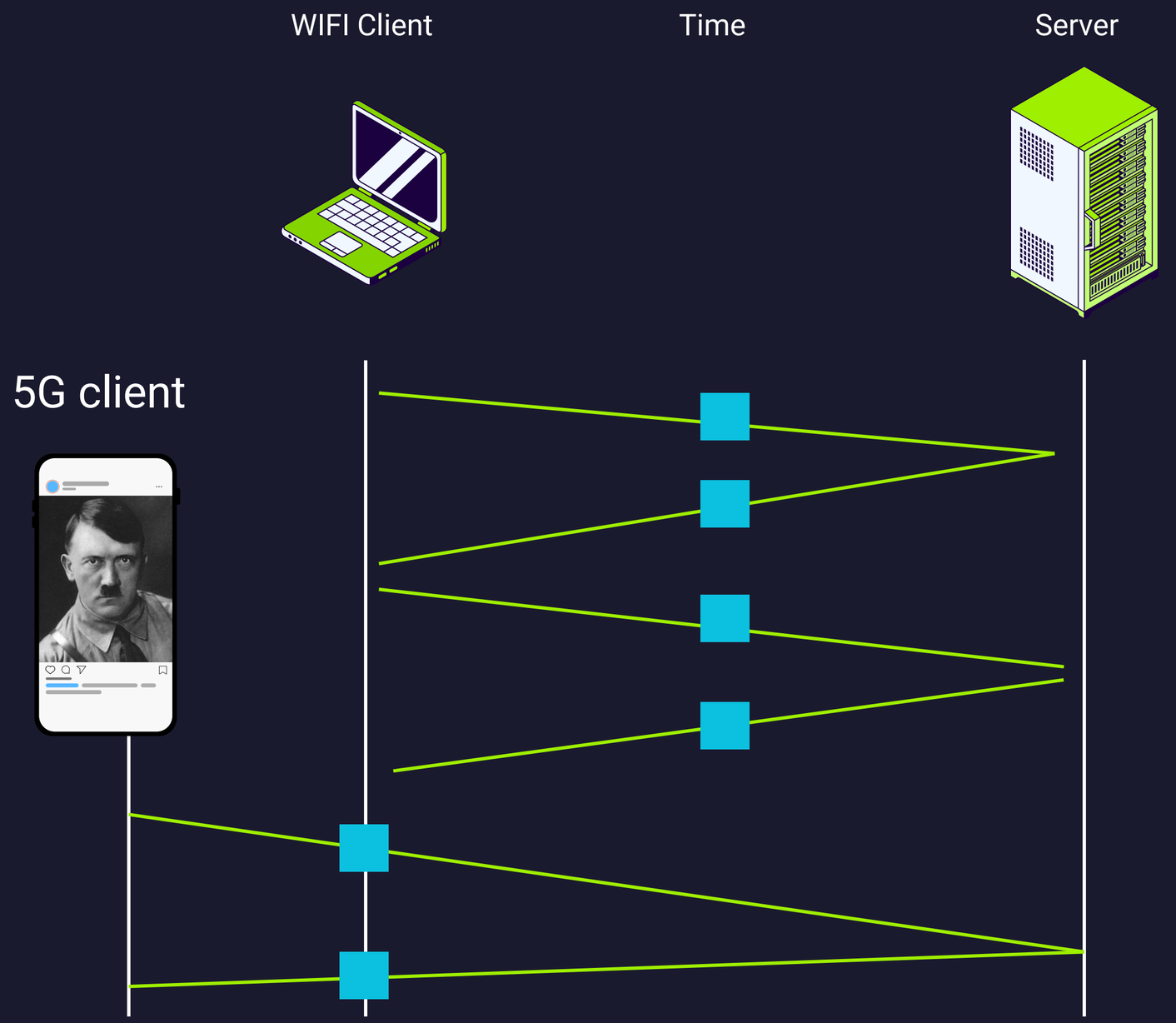
Provide security standards



Ph.D Mohammed Sahraoui

HTTP 3

The network is different but the connection id is the same.



Ph.D Mohammed Sahraoui

HTTP Request type

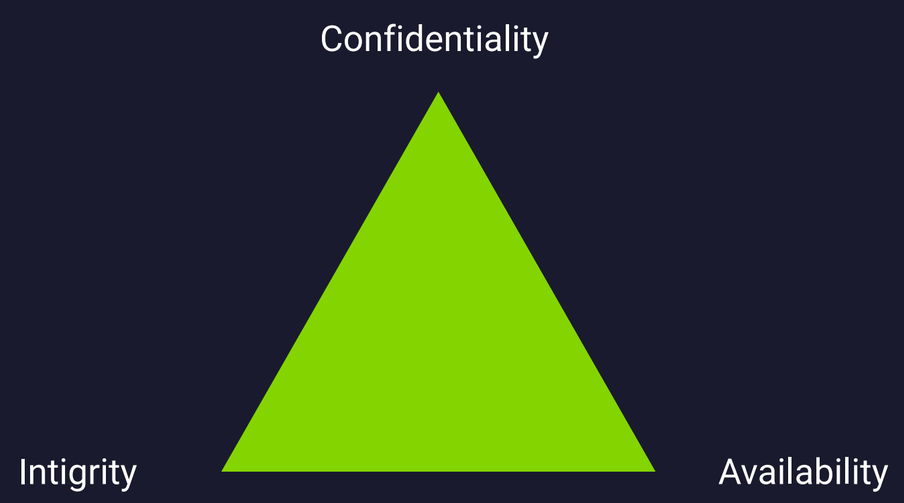
Method	Description	Typical Use Case	Idempotent?	Has Body?
GET	Retrieves data from the server.	Fetching a web page or API data	✅ Yes	❌ No
POST	Sends data to the server to create a	Submitting a form, uploading a file	❌ No	✅ Yes
PUT	Replaces or updates a resource entirely.	Updating user profile or settings	✅ Yes	✅ Yes
PATCH	Partially updates a resource.	Changing one field in a database record	✅ Yes	✅ Yes
DELETE	Removes a resource from the server.	Deleting a user or item	✅ Yes	Optional
HEAD	Same as GET but only retrieves headers (no	Checking if a resource exists	✅ Yes	❌ No
OPTIONS	Describes communication options	CORS preflight requests	✅ Yes	Optional
TRACE	Echoes the received request for	Debugging network paths	✅ Yes	❌ No
CONNECT	Establishes a tunnel to the server	Proxying SSL traffic	✅ Yes	❌ No

HTTP Status code

Code Range	Category	Example Codes & Meaning
100–199	Informational	100 Continue: Client should continue with request , 101 Switching Protocols
200–299	Success	200 OK: Request succeeded , 201 Created: Resource created , 204 No Content
300–399	Redirection	301 Moved Permanently ,302 Found, 304 Not Modified
400–499	Client Error	400 Bad Request , 401 Unauthorized , 403 Forbidden , 404 Not Found
500–599	Server Error	500 Internal Server Error , 502 Bad Gateway , 503 Service Unavailable

2.3 Introduction to TLS/SSL (HTTPS)

HTTPS, or Hypertext Transfer Protocol Secure, is an enhanced version of HTTP that incorporates security measures to protect data exchanged between a web browser and a server. It achieves this security through the use of TLS (Transport Layer Security) or SSL (Secure Sockets Layer) encryption protocols. TLS is the more modern and secure iteration, having succeeded SSL, and it ensures that information such as personal data, credit card details, and login credentials are transmitted securely over the internet. This encryption not only safeguards data integrity but also helps prevent eavesdropping and tampering by unauthorized parties.

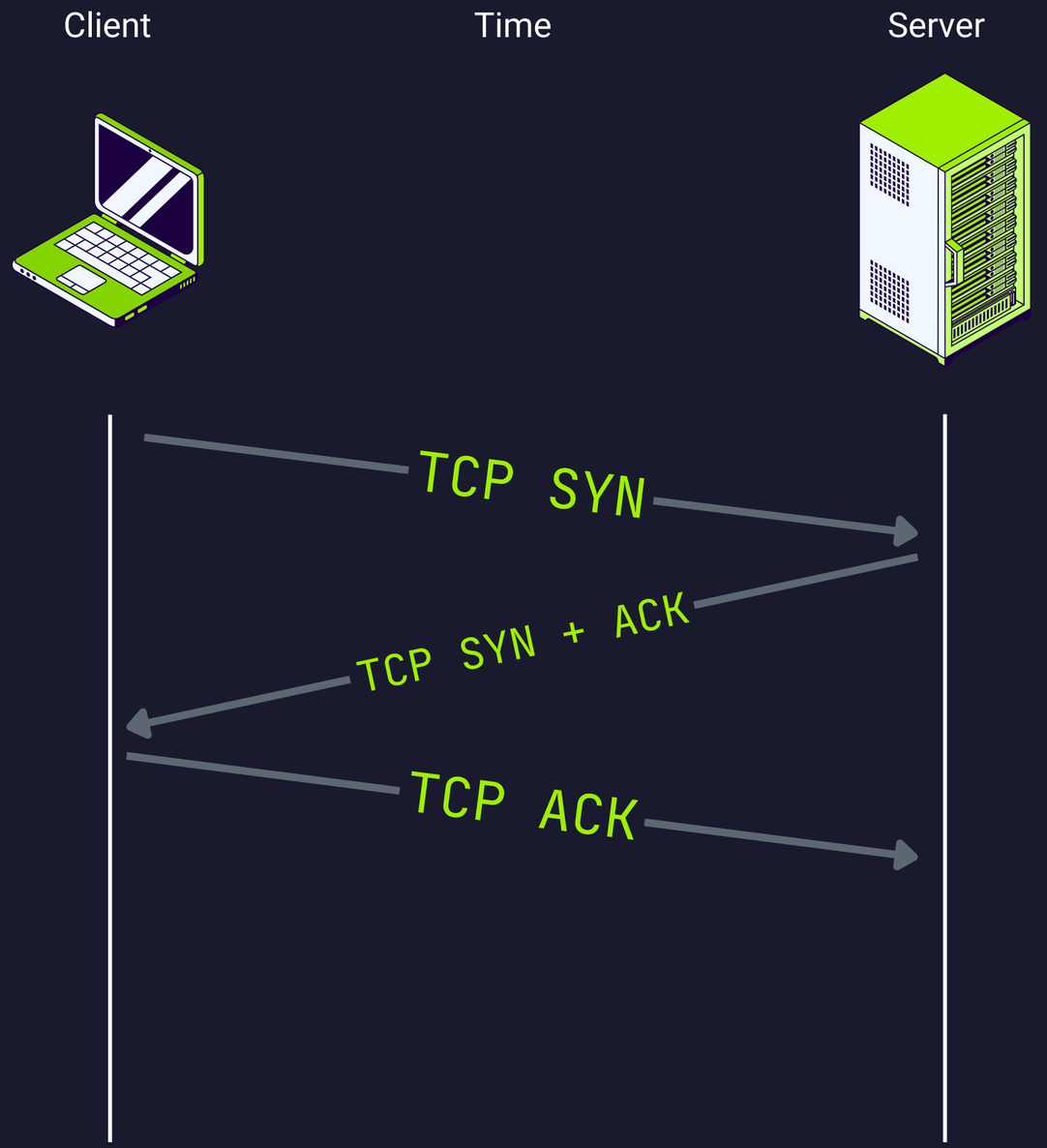


Ph.D Mohammed Sahraoui

Concept Visualization

TLS

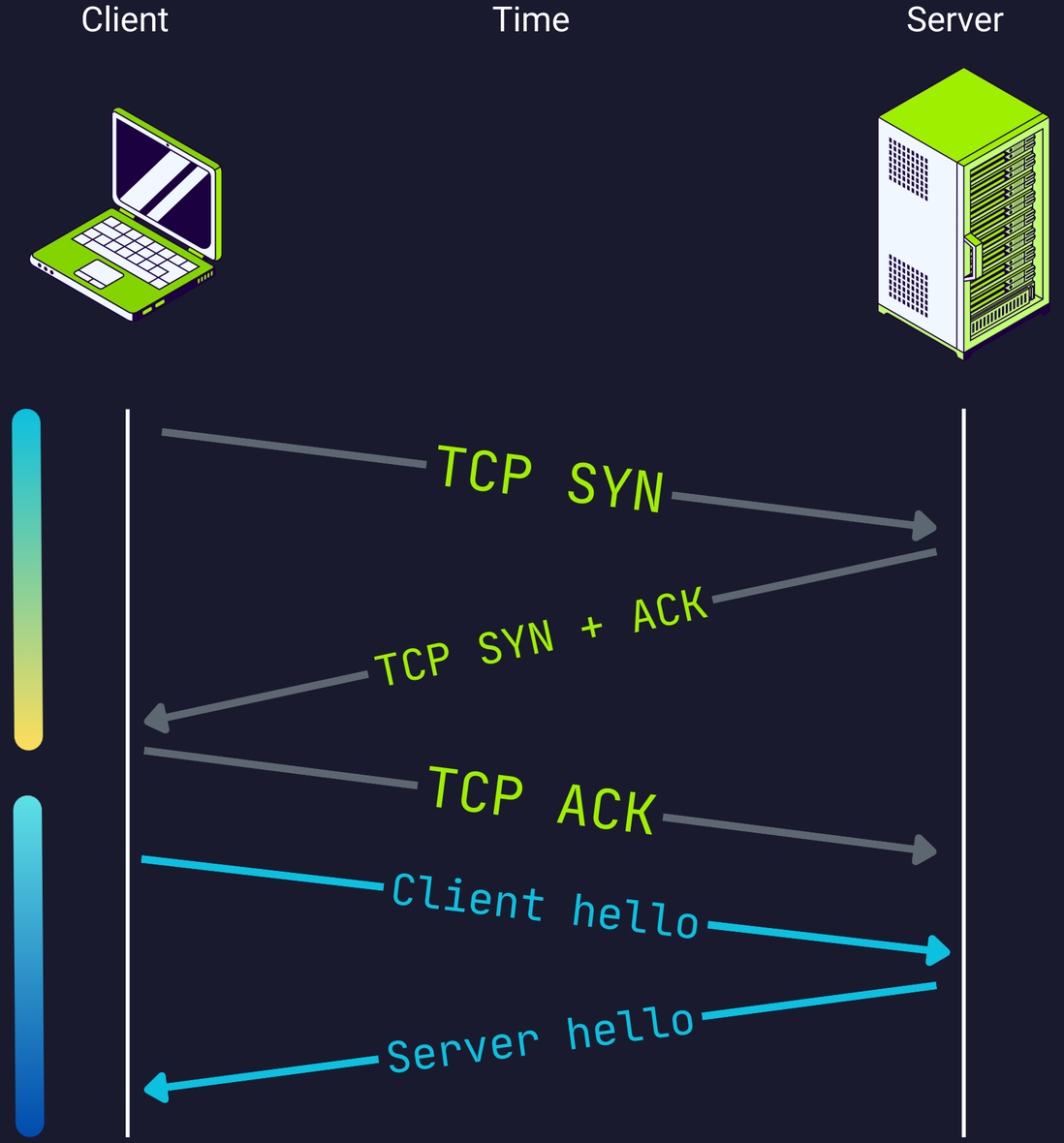
Encryption schemes



Ph.D Mohammed Sahraoui

TLS

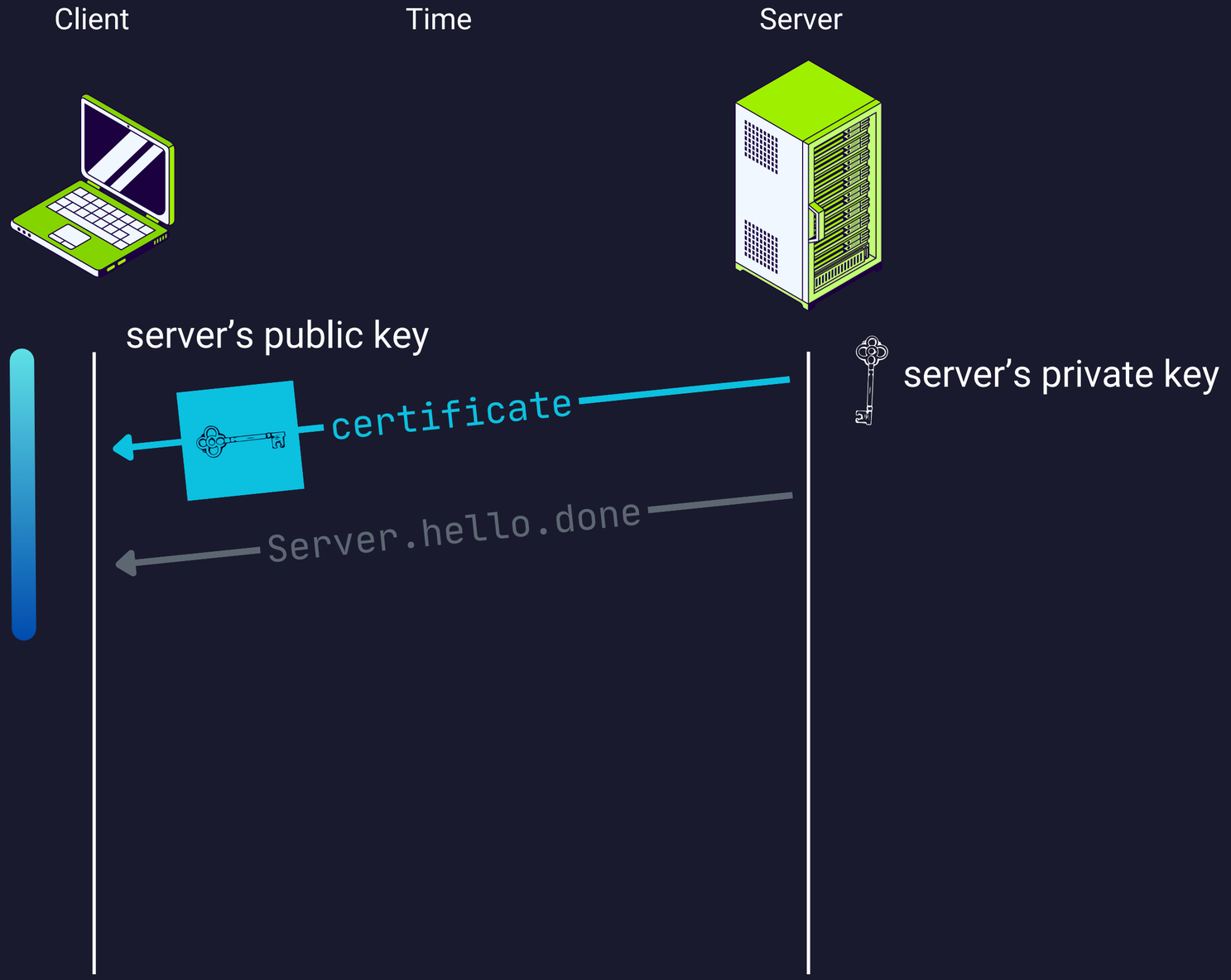
Encryption schemes



Ph.D Mohammed Sahraoui

TLS

Encryption schemes



Ph.D Mohammed Sahraoui

TLS

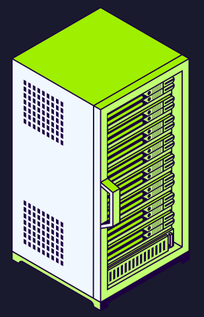
Encryption schemes

- shared secret key (session key) (sym encryption)
- shared secret key (encrypted session key)
- server's private key (RSA's decryption)
- server's public key (RSA's decryption)

Client

Time

Server

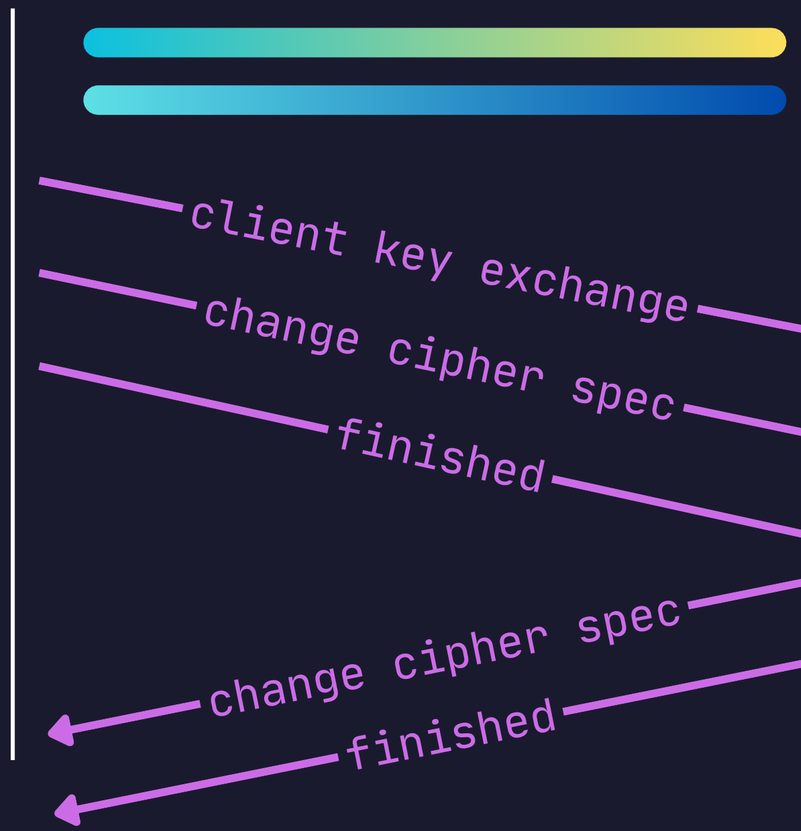


TLS handshake
Certification check

Encrypt it using the public server's public key using RSA

The `os.urandom()` function securely generates a random value that is 48 bytes in length.

Shared secret key



server's private key

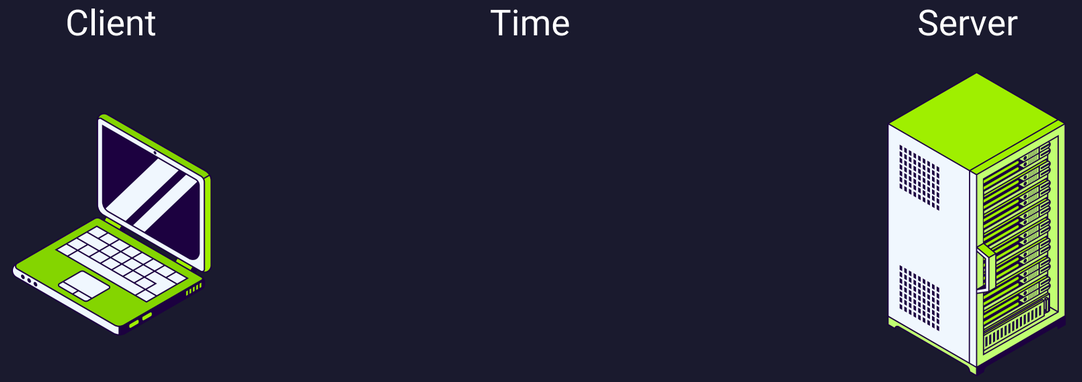
decrypt the shared secret key using server's private key

session key

Ph.D Mohammed Sahraoui

TLS

Encryption schemes



Ph.D Mohammed Sahraoui

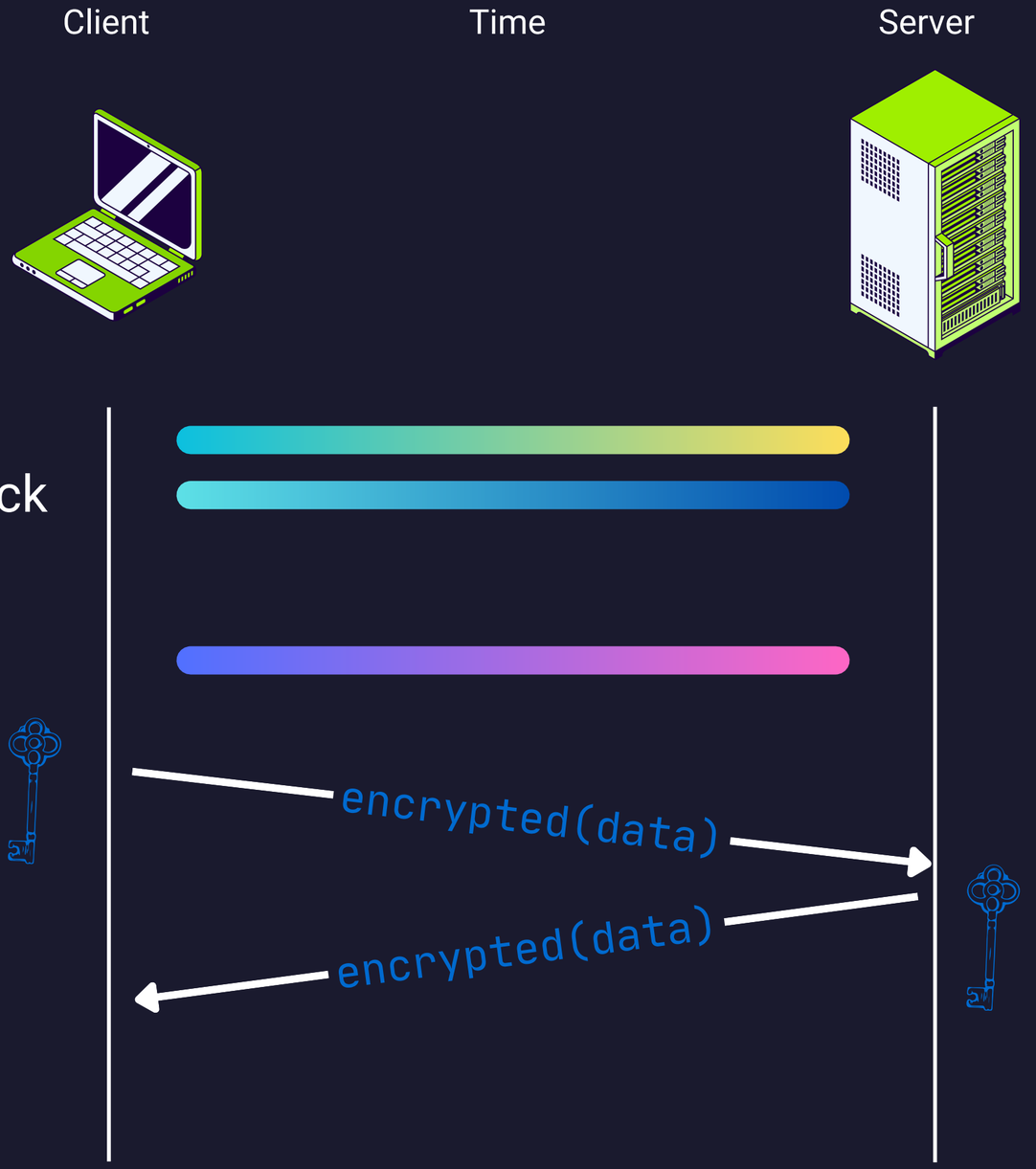
TLS

Encryption schemes

Ph.D Mohammed Sahraoui

TLS

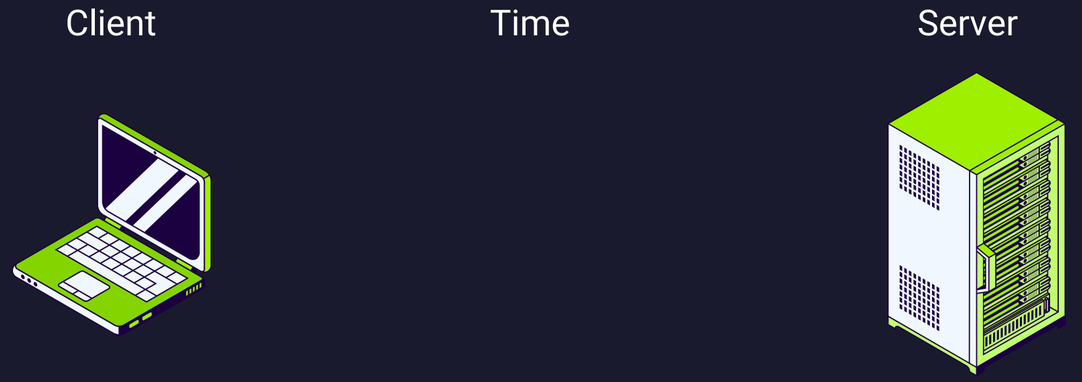
Encryption schemes



Ph.D Mohammed Sahraoui

TLS

Encryption schemes



Ph.D Mohammed Sahraoui

End.

Thank you for your time.

Ph.D Mohammed Sahraoui