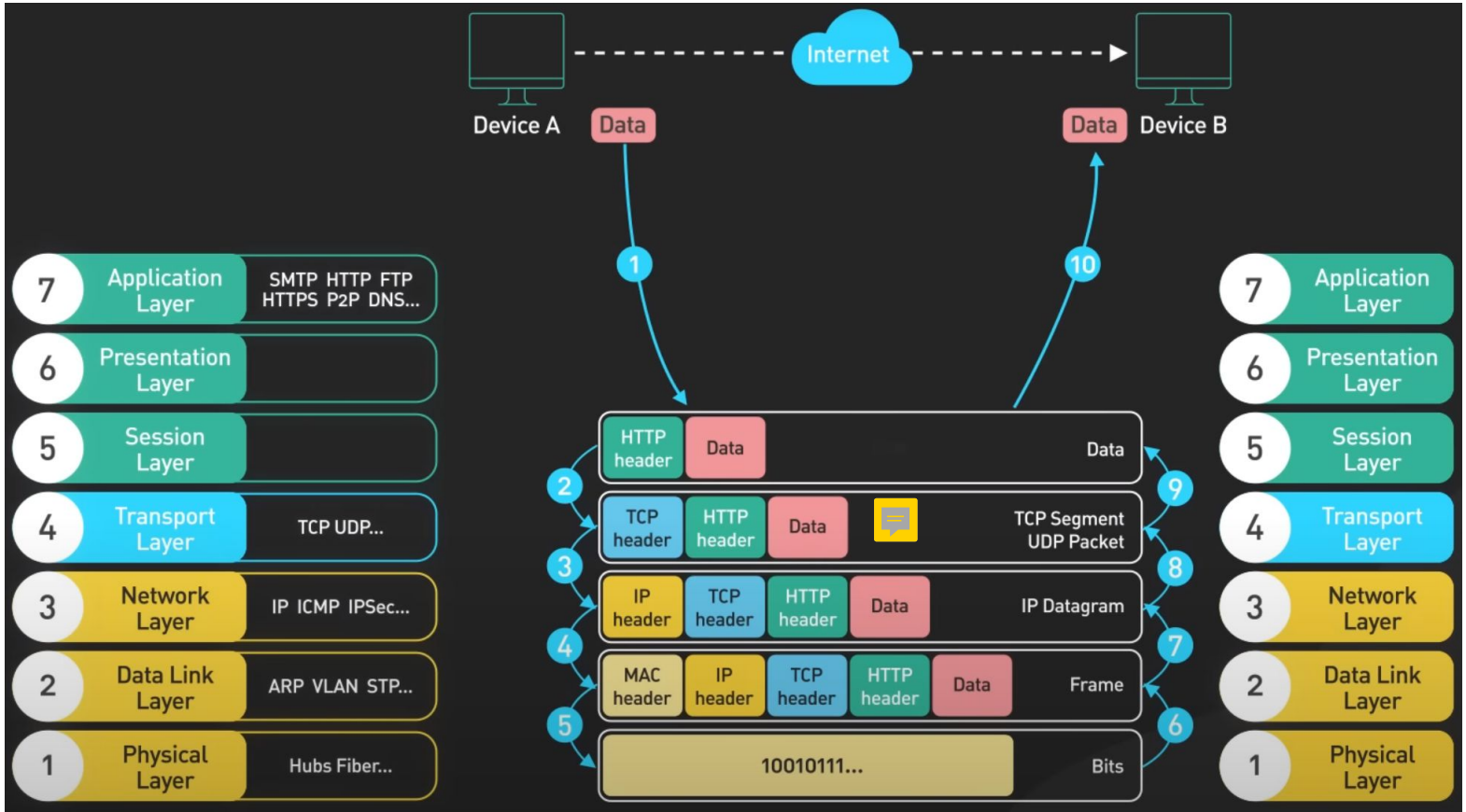
Go Sockets & RPCs

Lab 1

Outline

- OSI Model
- Sockets
- Remote Procedure Calls (RPCs)
- HTTP
- gRPCs
- Go Programming

OSI Model

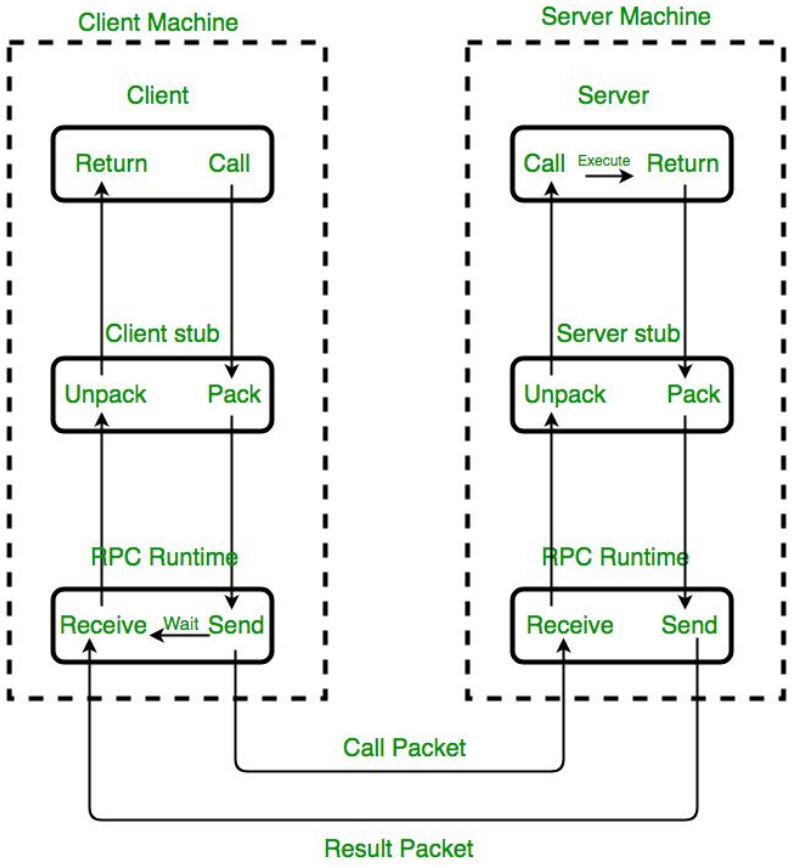

Internet
Device A
Data
Data
Device B
1
10
7 Application Layer
SMTP HTTP FTP HTTPS P2P DNS...
6 Presentation Layer
5 Session Layer
4 Transport Layer
TCP UDP...
3 Network Layer
IP ICMP IPSec...
2 Data Link Layer
ARP VLAN STP...
1 Physical Layer
Hubs Fiber...
HTTP header
Data
Data
2
9
TCP header
HTTP header
Data
TCP Segment
UDP Packet
3
8
IP header
TCP header
HTTP header
Data
IP Datagram
4
7
MAC header
IP header
TCP header
HTTP header
Data
Frame
5
6
10010111...
Bits
7 Application Layer
6 Presentation Layer
5 Session Layer
4 Transport Layer
3 Network Layer
2 Data Link Layer
1 Physical Layer

Sockets

- API for transport layer protocols (TCP/UDP)
- UDP: unreliable datagram
- TCP: reliable, byte stream-oriented

Remote Procedure Calls

- Enables users to work with remote procedures as if the procedures were local (*called procedure need not exist in the same address space as the calling procedure*)
- An implementation of RPC can be done over basically any network transport (e.g. TCP, UDP, cups with strings)

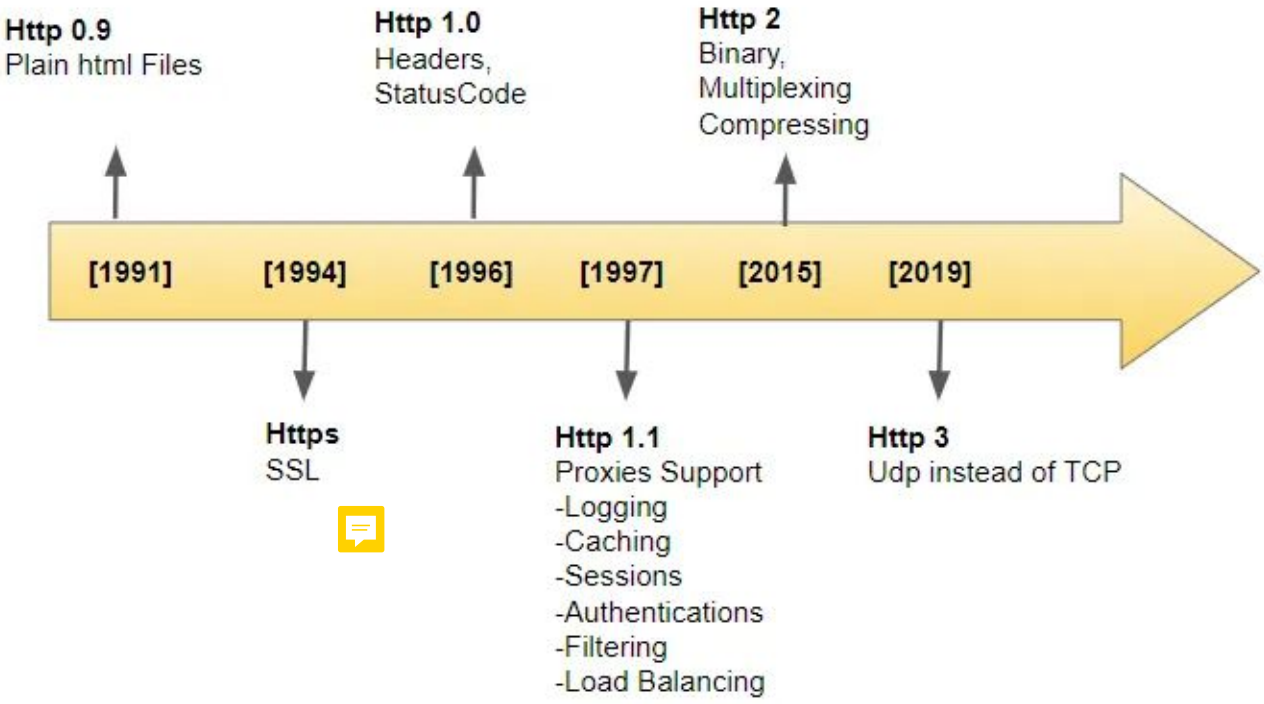


Implementation of RPC mechanism

HTTP

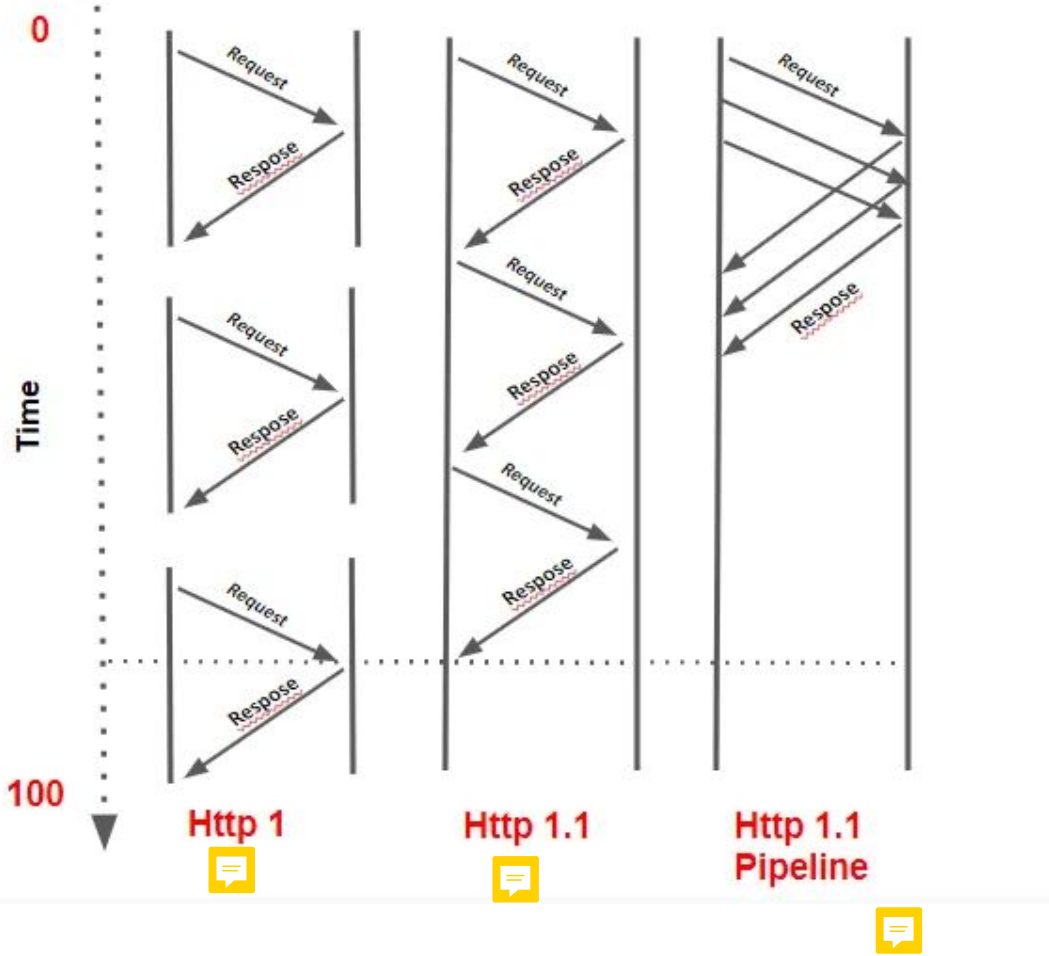
- The Hypertext Transfer Protocol (HTTP) is a stateless application-level protocol for distributed, collaborative, hypermedia information systems.
- Simple
- Extensible
- Stateless
- Reliable Connections
- Caching, Sessions
- Authentications

HTTP Evolution



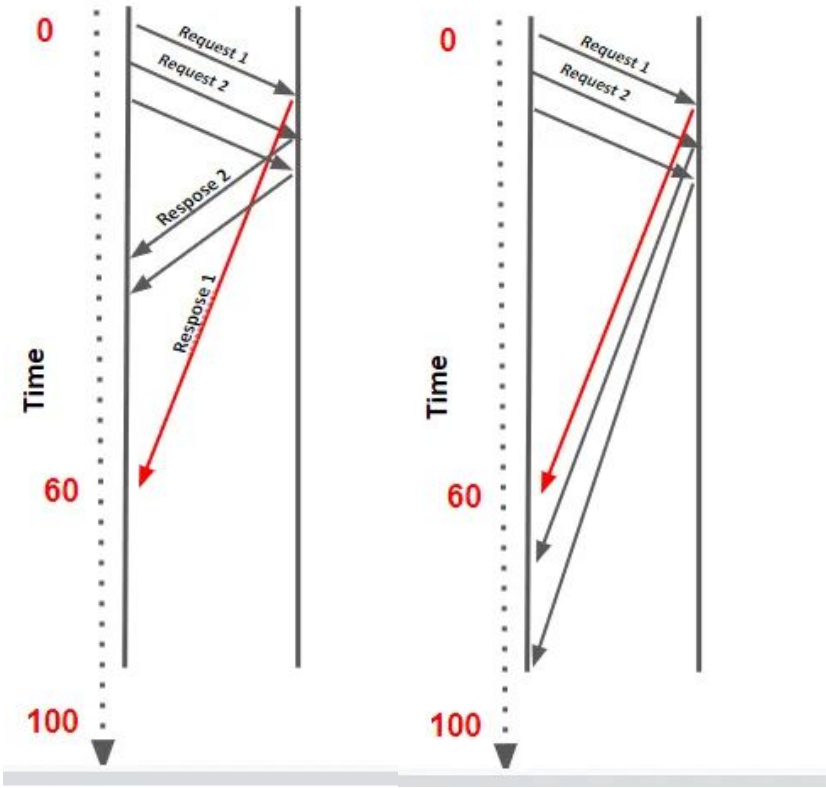
HTTP 1

- If we look at the “Http1.1 Round Trip” figure, the time latency was solved. That’s perfect. Especialy in pipeline, look like ideal solution to high load.
- This ordering of requests and responses is required otherwise there is no way for the client and the server to figure out the corresponding requests and responses.
- But what happen if one request’s response time higher than others.



HTTP 1.1 Head Of Line Blocking

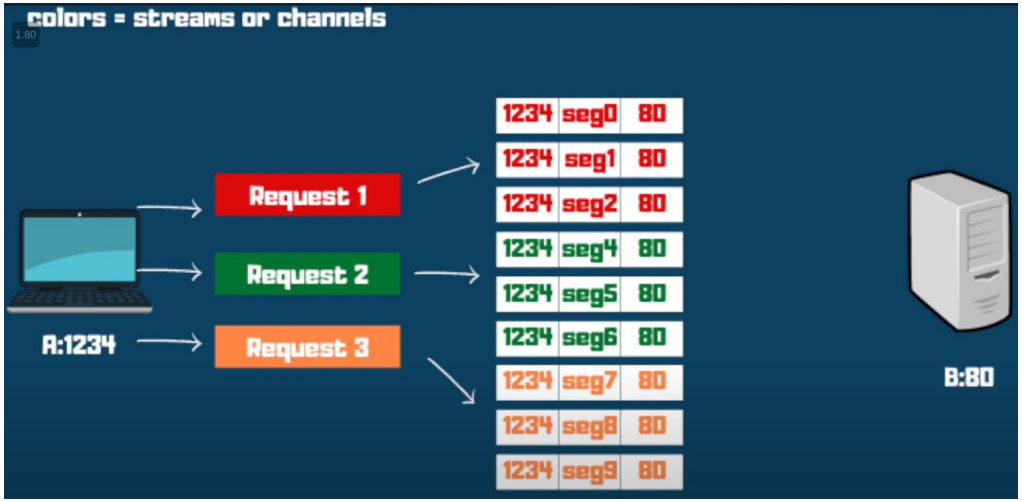
Even though request 2 & 3 finishes execution before request 1, their responses can’t be sent before request 1.



HTTP2 to the Rescue

With this, since each frame has a stream id associated with it, it removes the requirement to process HTTP requests and responses in order by the clients and servers.

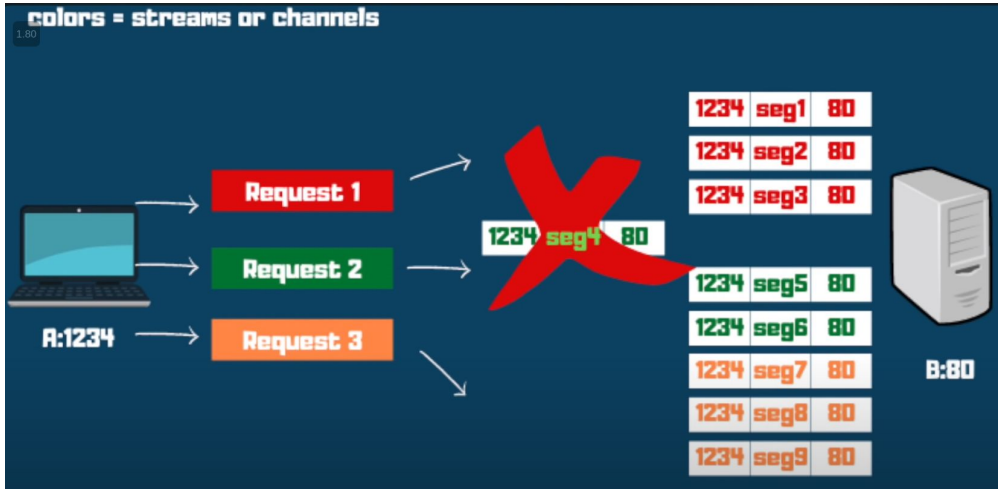
The HOL blocking issue is thus resolved at the HTTP layer, but it now moves to the TCP layer.



HTTP2 HOL Blocking

To understand why that is the case, recall that every TCP packet carries a unique sequence number when put on the wire, and the data must be passed to the receiver in order.

If one of the packets is lost en route to the receiver, then all subsequent packets must be held in the receiver's TCP buffer until the lost packet is retransmitted and arrives at the receiver.



gRPCs

- Designed to be fast, efficient, and scalable, which makes it ideal for use in large-scale distributed systems
- Implements traditional RPC with several optimizations. For instance, gRPC uses Protocol Buffers and HTTP 2 for data transmission.
- Protocol Buffers binary serialization format for efficient data exchange.
- Supports a wide range of programming languages.

Go Programming

References

- https://www.youtube.com/@hnasr
- https://www.youtube.com/@pragma-ar
- https://www.youtube.com/watch?app=desktop&v=gnchfOojMk4
- https://medium.com/@eyupcanarslan/who-shot-rest-lets-discover-grpc-176667e0e0a2
- https://engineering.cred.club/head-of-line-hol-blocking-in-http-1-and-http-2-50b24e9e3372
- https://aws.amazon.com/compare/the-difference-between-grpc-and-rest/
- ChatGPT helped in code examples.