DIFFERENCE BETWEEN HTTP1.1 VS HTTP2

NOVEMBER 17, 2023

GOWSALYA SANJAI KUMAR

FSD54WD-T

DIFFERENCE BETWEEN HTTP1.1 vs HTTP2

What is HTTP?

HTTP stands for Hypertext Transfer Protocol. It is the foundation of the World Wide Web and is used by browsers to load web pages. A typical example is when your browser sends an HTTP request to a web server after entering an URL. The HTTP command then provides an HTTP response to the web server with the webpage's contents.

Difference between HTTP1.1 vs HTTP2

Header:

	Header Format, Compression and Transmission
HTTP/1.1	➢ ASCII /Plaintext ➢ No compression ➢ Header field names - case insensitive
HTTP/2	➢ HPACK compression of header into block ➢ Breaks header block into frames for transmission ➢ Huffman encoding + Static table of commonly used header fields + Dynamic table with fields specific to the session ➢ All field names lower case and request line is split into separate pseudo-header fields :method, :scheme, :authority, and :path.

Message Transmission Format - Framing

	Message Format
HTTP/1.1	➢ ASCII /Plaintext ➢ Header Section ➢ Message Body ➢ Separated by empty line
HTTP/2	➢ Frames ➢ Headers Frame ➢ Data Frame - Payload

DIFFERENCE BETWEEN HTTP1.1 vs HTTP2

Transport and Security Mechanisms

	Transport Mechanism	Security
HTTP/1.1	➢ TCP Session	➢ Transport Layer Security(TLS) I. TLS 1.2 II. Previously → SSL ➢ Hypertext Transfer Protocol Secure (HTTPS) ➢ Bi-directional encryption between client and server
HTTP/2	➢ TCP Session	➢ Same as in HTTP/1.1 i.e. optionally runs over TLS for encrypted connection

Connection Management: Estab, Persistence & Closure

	Connection Establishment	Persistence	Closure
HTTP/1.1	➢ Client initiates TCP connection ➢ Multiple simultaneous TCP connections allowed	➢ Persistent By default ➢ Recipient determines the status based on protocol version of tmost recently received message or on connection header	➢ "close" connection header option to signal closing init ➢ sender or receiver ➢ Premature closing, ➢ re-open automatically, once
HTTP/2	➢ Client initiates TCP connection ➢ Single connection per host-port pair for each server, ➢ Multiple streams can be run	➢ Persistent By default ➢ Can be closed if idle	➢ Connection can be closed if idle ➢ Endpoints should send GOAWAY message to signal initiating graceful closing ➢ Can close without GOAWAY if misbehaving peer

Message Ordering, Multiplexing & Concurrency

	Message Ordering	Multiplexing	Concurrency
HTTP/1.1	➢ Queued ➢ Head-of-line blocking i.e. one request serviced at a time	➢ Not Multiplexed	➢ Parallel sessions can be run via parallel independent but simultaneous connections
HTTP/2	➢ Absolute ordering of frames spanning across all streams ➢ Each stream has integer identifier ➢ Frame sending order determines receive order.	➢ Multiplexed across streams	➢ Several streams can be open concurrently and frames from multiple streams can be interleaved ➢ "stream" is an independent, bidirectional sequence of frames exchanged between the client and server

DIFFERENCE BETWEEN HTTP1.1 vs HTTP2

Flow Control, Congestion Control, Prioritization

	Flow Control	**Congestion Control**	**Prioritization**
HTTP/1.1	➢ No flow control ➢ Relies on TCP	➢ No congestion control	➢ No prioritization
HTTP/2	➢ Flow control provided for entire connection i.e. across streams but not per stream ➢ Only data frames subject to flow control ➢ Any algorithm	➢ Provided by TCP	➢ Client can assign priority status for a new stream via the HEADERS frame ➢ Can update it later using a PRIORITY frame

Cross-Version Compatibility - Upward & Downward

	Upgrading	**Read/Reply Lower Version**
HTTP/1.1	➢ Start a connection using HTTP/1.1 ➢ Request upgrade to HTTP/2 using upgrade header ➢ Can only upgrade to h2c ➢ → HTTP/2 Cleartext" ➢ Initiated by client but a server can require it	➢ compatible with HTTP/0.9, 1.0 ➢ can recognize the request line and any valid request ➢ respond appropriately with a message in the same version used by the client. ➢ recognize the status line in HTTP/1.0
HTTP/2	➢ No upgrade mechanism	➢ Fully compatible with HTTP/1.1,

Conclusion:

- Hypertext Transfer Protocol (HTTP) has undergone numerous changes since it was first adopted → Now multiple versions of HTTP exist
- Each version filling in gaps that existed in the previous one
- HTTP/0.x got the core concept up and running—a stateless application-level protocol for distributed, collaborative, hypertext information exchange.
- HTTP/1.x solved details such as the need for persistent connections and name-based virtual hosts. Security Introduced here SSL → TLS
- HTTP/2 introduced binary message framing, multiplexing and other extensions to optimize performance
- HTTP/3—the latest version— adds per-stream multiplexing and flow control plus packet-level security → adds reliability, reduces latency and improves security