HTTP & its Versions:

While the invention of World Wide Web, Tim Berners-Lee also invented the communication protocol (the first version of HTTP – Hyper Text Transfer Protocol) to be used in WWW.

## HTTP/0.9:

* It was a Simple, One-Line Protocol which used only GET method followed by path to the resource as request.
* As a response, It has no status/error codes and can transmit only html documents (hypertexts).

Eg. I/p: GET /index.html

O/p: <html>Contents…</html>

## HTTP/1.0:

* This version is called “Building Extensibility” – HTTP/0.9 + new capabilities.
* Methods supported were GET, HEAD, POST.
* Request format is Method type followed by path to the resource and versioning of the HTTP.
* As a response, we get the status code to understand the success/failure of the request.
* The notion of HTTP headers (metadata transmitted) have been introduced where the ability to transfer other documents (Eg. Media, stylesheets, scripts) has been added.

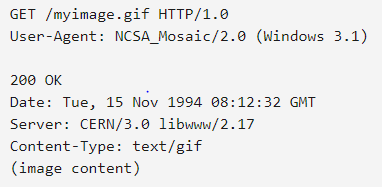
Eg. I/p: GET /mypage.html HTTP/1.0

User-Agent: NCSA\_Mosaic/2.0 (Windows 3.1)

O/p: 200 OK

Date: Tue, 15 Nov 1994 08:12:31 GMT

Server: CERN/3.0 libwww/2.17

Content-Type: text/html

<HTML>

A page with an image

<IMG SRC="/myimage.gif"> 🡪

</HTML>

For both the versions above, the connection would terminate immediately after the response. To retrieve a single document (embedded with multiple resources), multiple connection requests are made(i.e for each request ). This led to the performance degradation which was the major issue which led to development of HTTP/1.1.

## HTTP/1.1 – The standardized Protocol:

* This version is most commonly used and stabilized version for over 15 years.
* Methods supported are GET, HEAD, POST, PUT, DELETE, TRACE, OPTIONS.
* Introduced more performance improving functionalities:
  + Persistent(long-lived) and pipelined(allows client to send a second request before the answer for the first one is fully transmitted, lowering the latency of the communication) connections.
  + chunked transfers( responses are made as series of chunks where total size of the response may not be known until the request has been fully processed).
  + content negotiations (eg. Language, encoding, etc. which allows client and server to agree on most adequate content to exchange).
  + virtual hosting (a server with a single IP Address hosting multiple domains)
  + cache support - faster response and great bandwidth savings.

HTTP/1.1 – used for secure transmission(initially, SSL(encrypted transmission layer over TCP/IP stack) which later transformed to TLS is used to guaranty the authenticity of the messages b/w client and server); used for complex applications(WebDAV(Web Distributed Authoring and Versioning)-> RESTful APIs);

HTTP/1.1 was far from optimal at using the network i.e. it was wasting the bandwidth. And using multiple TCP channels between a certain client and a single server does not speed up the communication, it was only a workaround in HTTP/1.1 to partially mitigate the choking effect of slow resources.

## HTTP/2 – Protocol for greater performance:

Webpages that were developing became complex which demanded more performance. For HTTP/1.1 – Request has to be in-order; Parallel connections can be used – created more traffic and complexity for complex webpages; Pipelining – has become resource burden in web development

So, Google demonstrated an alternative way of exchanging data between client and server, by implementing an experimental protocol SPDY. Defining an increase in responsiveness, and solving the problem of duplication of data transmitted, SPDY served as the foundations of the HTTP/2 protocol.

* Binary protocol – HTTP/2 essentially wraps HTTP/1 in such a binary protocol.
  + HTTP/1.x is a textual protocol(stream of characters rather than being broken to separate pieces/frames). Whereas, HTTP/2.0 has highly structured format(FrameHeader[length+frameType]+Payload[HTTPheaders&BodyPayload]) where HTTP messages are formatted into packets (called frames) and where each frame is assigned to a stream and the message can be sent in one/more frames.
* Improved optimization techniques can be implemented.
* Multiplexed protocol – Parallel request can be handled in same connection irrespective to the order of requests processed(Req1,Req2,Req3 -> Req2,Req1,Req3)
  + In HTTP/1.1, the browser downloads the resources one after the other in separate TCP channels. The number of the TCP channels the server or, for that matter, the client can handle is finite, therefore, the browsers limit themselves not to open more than four TCP channels. It means that while four content elements for the page are downloading, the other elements wait in a queue to be downloaded.
  + <https://freecontent.manning.com/animation-http-1-1-vs-http-2-vs-http-2-with-push/>
* Compresses header – Up until HTTP/1.x, the headers were sent over repeatedly for every request leading to a lot of duplicate data being sent uncompressed and causing inefficient use of bandwidth and introducing delays in page load time. Therefore, Compressing headers(Headers needed to be sent only 1 time in an connection)reduces their size in terms of numbers of bytes that are transmitted during the connection.
  + HTTP/2 deploys the HPACK format for header compression, the pages will load faster with better interactive response time
* Server push – lets the server to push content to the client before the client requests the particular content with preconditions applied.

Although HTTPS is secure by its design, SSL/TLS takes time(1-2 seconds) to establish a connection then rather slows down the startup performance of website.

## HTTP/3.0:

The next major version of HTTP, HTTP/3, will use [QUIC](https://developer.mozilla.org/en-US/docs/Glossary/QUIC) instead [TCP](https://developer.mozilla.org/en-US/docs/Glossary/TCP)/[TLS](https://developer.mozilla.org/en-US/docs/Glossary/TLS) for the transport layer portion.