**DIFFERENCES BETWEEN HTTP VERSIONS**

|  |  |  |  |
| --- | --- | --- | --- |
| Differentiator | HTTP/1.0 | HTTP/1.1 | HTTP/2 |
| Year | 1991 | 1997 | 2015 |
| Key Features | For every TCP connection there is only one request and one response. | It supports connection reuse i.e. for every TCP connection there could be multiple requests and responses, and pipelining where the client can request several resources from the server at once. However, pipelining was hard to implement due to issues such as head-of-line blocking and was not a feasible solution. | Uses multiplexing, where over a single TCP connection resources to be delivered are interleaved and arrive at the client almost at the same time. It is done using streams which can be prioritized, can have dependencies and individual flow control. It also provides a feature called server push that allows the server to send data that the client will need but has not yet requested. |
| Status Code | Can define 16 status codes; the error prompt is not specific enough. | Introduces a warning header field to carry additional information about the status of a message. Can define 24 status codes, error reporting is quicker and more efficient. | Underlying semantics of HTTP such as headers, status codes remains the same. |
| Authentication Mechanism | Uses basic authentication scheme which is unsafe since username and passwords are transmitted in clear text or base64 encoded. | It is relatively secure since it uses digest authentication, NTLM authentication. | Security concerns from previous versions will continue to be seen in HTTP/2. However, it is better equipped to deal with them due to new TLS features like connection error of type Inadequate\_Security. |
| Caching | Provides support for caching via the If-Modified-Since header. | Expands on the caching support by using additional headers like cache-control, conditional headers like If-Match and by using entity tags. | HTTP/2 does not change much in terms of caching. With the server push feature if the client finds the resources are already present in the cache, it can cancel the pushed stream. |
| Web Traffic | HTTP/1.1 provides faster delivery of web pages and reduces web traffic as compared to HTTP/1.0. However, TCP starts slowly and with domain sharding (resources can be downloaded simultaneously by using multiple domains), connection reuse and pipelining, there is an increased risk of network congestion. | | HTTP/2 utilizes multiplexing and server push to effectively reduce the page load time by a greater margin along with being less sensitive to network delays. |
| Header Compression | Headers are sent on every request leading to a lot of duplicate data being sent uncompressed across the wire. | | Header compression is included by default in HTTP/2 using HPACK. |
| Performance Optimization | Provides support for caching to deliver pages faster. | Spriting, concatenating, inlining, domain sharding are some of the optimizations used as a workaround to the ‘six connections per host’ rule. | Removes the need for unnecessary optimization hacks. |
| Protocol Type | Text based protocol that is in the readable form. | | It is a binary protocol (HTTP requests are sent in the form of 0s and 1s). Needs to be converted back from binary in order to read it. |
| Security | SSL is not required but recommended. Digest authentication used in HTTP1.1 is an improvement over HTTP1.0. HTTPS uses SSL/TLS for secure encrypted communication. | | Though security is still not mandatory, it is mostly encrypted (though it is not enforced) since almost all clients require traffic to be encrypted. It also has some minimum standards, such as minimum key size for encryption. TLS 1.2 etc. |

**DIFFERENCES BETWEEN GET AND POST**

|  |  |
| --- | --- |
| **GET** | **POST** |
| GET is used to request data from a specified resource | POST is used to send data to a server to create/update a resource. |
| GET requests can be cached | POST requests are never cached |
| GET requests remain in the browser history | POST requests do not remain in the browser history |
| GET requests can be bookmarked | POST requests cannot be bookmarked |
| GET requests have length restrictions | POST requests have no restrictions on data length |
| GET requests are only used to request data (not modify) | POST is used to update a resouce |