|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| s.no | HTTP1.1 | **Vs** | | HTTP2 |
| 1.  2.  3.  4.  5.  6.  7.  8.  9.  10. | Uses a single connection for each resource, leading to head-of-line blocking, where subsequent requests wait for earlier ones to complete.  Uses a text-based protocol, which is human-readable but comes with higher overhead due to the verbosity of textual data.  Headers are sent in plain text for each request and response, resulting in redundant data transmission.  Does not have built-in mechanisms for specifying the priority of resources, leading to suboptimal page loading strategies.  Requires the client to initiate each request, leading to a delay in loading dependent resources.  Requires multiple connections for parallelization, each incurring its own overhead.  Lacks a formal mechanism for expressing dependencies between resources.  Has limited support for flow control, which can lead to inefficiencies in handling resource delivery.  Relies on TCP for error recovery, which can lead to retransmission delays and increased latency.  Maintains compatibility with older web systems and infrastructure. | | Employs multiplexing, allowing multiple requests and responses to be processed concurrently over a single connection, eliminating head-of-line blocking.  Adopts a binary protocol, reducing overhead and enabling more efficient parsing and processing of data by both browsers and servers.  Implements header compression, reducing the size of headers through the use of a dynamic table that stores frequently used header fields, thereby minimizing overhead.  Introduces resource prioritization, allowing the client to indicate the importance of different resources, ensuring critical assets are loaded first for improved performance.  Supports server push, enabling servers to proactively send resources to the client's cache before they are explicitly requested, reducing latency and improving page load times.  Promotes a single, multiplexed connection, reducing the need for multiple connections and associated overhead.  Introduces stream dependencies, allowing the client to specify the order in which resources should be processed, improving overall page load efficiency.  Implements more robust flow control mechanisms, ensuring optimal resource transmission between the client and server.  Uses its own error handling mechanisms, reducing the impact of network errors and improving overall reliability.  Designed to be backward-compatible with HTTP/1.1, allowing a smooth transition for systems and applications that have not yet adopted the new protocol. | |
|  |  | |  | |