HTTP/1.1:

1. \*\*Request Limitations:\*\*

- \*\*Multiplexing:\*\* In HTTP/1.1, each request/response operates in a serialized manner, meaning only one request can be processed at a time per TCP connection.

- \*\*Header Overhead:\*\* The protocol requires sending headers with each request and response, leading to higher overhead and latency.

2. \*\*Performance Challenges:\*\*

- \*\*Latency:\*\* Due to the limit on concurrent connections, websites with many resources (images, scripts, stylesheets) faced increased latency as each resource required a separate connection.

- \*\*Resource Inefficiency:\*\* Redundant data transmission and high-latency overhead affected overall performance, especially for complex websites.

3. \*\*Compression and Security:\*\*

- \*\*No Built-in Compression:\*\* Compression was optional in HTTP/1.1, causing larger data payloads and slower transmission.

- \*\*Security Concerns:\*\* Security features like domain sharding (splitting resources across multiple domains) for parallel downloads also posed security risks.

HTTP/2:

1. \*\*Multiplexing and Stream Prioritization:\*\*

- \*\*Multiplexing:\*\* HTTP/2 introduced multiplexing, allowing multiple requests and responses to be sent concurrently over a single TCP connection. This significantly reduced latency.

- \*\*Stream Prioritization:\*\* Prioritizing streams ensured critical resources (like CSS or JavaScript) loaded first, enhancing overall page load times.

2. \*\*Header Compression:\*\*

- \*\*Header Compression:\*\* HTTP/2 uses header compression (HPACK) to reduce overhead by compressing repeated headers, improving efficiency.

3. \*\*Server Push:\*\*

- \*\*Server Push:\*\* A groundbreaking feature of HTTP/2, it enables servers to push multiple responses (resources) to a client before the client requests them. This optimizes resource loading, particularly for interconnected resources.

4. \*\*Binary Protocol:\*\*

- \*\*Binary Framing Layer:\*\* HTTP/2 operates on a binary protocol instead of plain text, making it more efficient in parsing and reducing errors compared to the textual format of HTTP/1.1.

5. \*\*Security and Compatibility:\*\*

- \*\*TLS Usage:\*\* Though not mandatory, HTTP/2 generally operates over TLS (Transport Layer Security), emphasizing security for data transmission.

- \*\*Backward Compatibility:\*\* HTTP/2 maintains compatibility with HTTP/1.1, allowing for a graceful transition for systems and websites.

Conclusion:

HTTP/2's improvements in multiplexing, header compression, and server push alleviate many of the performance bottlenecks present in HTTP/1.1. It enables faster, more efficient data transmission, leading to improved user experiences with quicker page loads and reduced latency.

As technology continues to evolve, protocols like HTTP/2 set the stage for even more efficient and secure communication between servers and clients on the web.