Write a blog on Difference between HTTP1.1 vs HTTP2

HTTP2 is much faster and more reliable than HTTP1. HTTP1 loads a single request for every TCP connection, while HTTP2 avoids network delay by using multiplexing.

HTTP is a network delay sensitive protocol in the sense that if there is less network delay, then the page loads faster. However, an impressive increase in network bandwidth only slightly improves page load time. This is key to understanding the differences in performance efficiencies between the different versions of HTTP. Back in the day when people used dial up modems web pages were simple and it was the actual data transfer between the server and the client that contributed towards the largest chunk of the page load time. Today the actual downloading of resources from server takes a negligible portion of the total page load time due to the tremendous increase in bandwidth availability. It is the time taken to establish the TCP connection and making requests that impacts performance. It was initially recommended to use only two connections per hostname but today most browsers use six connections per hostname. When we talk about http vs http2 in terms of performance it is important to note that a lot of performance optimizations adopted by HTTP/1.1 introduced complexities in terms of developmental efforts as well as network congestion that HTTP/2 attempts to address.

The table below points out the differentiating factors between http2 vs http1:

Header CompressionHeaders 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 OptimizationProvides 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 TypeText 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.SecuritySSL 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.

Header CompressionHeaders 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 OptimizationProvides 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 TypeText 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.SecuritySSL 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.

|  |  |  |  |
| --- | --- | --- | --- |
| 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 CompressionHeaders 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. SecuritySSL 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.

## How to Implement HTTP/2 on Your Website

Since using HTTP/2 is an invisible process when correctly implemented, your website may already be using it without your realization. There is an easy way to check this:

* Open the web developer tool on the web browser (like Firefox).
* Under the network tab, select any of the resources and check the version number under the headers tab.

While HTTP/2 does not mandate the use of SSL, it is crucial to install an SSL certificate because the leading browsers, including Firefox and Chrome, have decided to implement HTTP/2 only over TLS (HTTPS). In order to enable HTTP/2 it is essential to get an SSL/TLS certificate and make every page on the website https.

At the web server level, it could be as simple as a software update, for example, Apache began support for HTTP/2 in version 2.4.17.

## Adoption Of HTTP/2

HTTP/2 penetration on the client side is more than 70% as most major browsers support HTTP/2 and on the server side we have major tech giants like Google, Facebook, Nginx, etc. who have their own servers supporting HTTP/2. The adoption rate, [according to W3Techs](https://w3techs.com/technologies/details/ce-http2), is currently around 29% globally.

### Related Posts:

**Objects and its internal representation in Javascript**

Object:

In JavaScript, an object is a standalone entity, with properties and type. Compare it with a cup, for example. A cup is an object, with properties. A cup has a color, a design, weight, a material it is made of, etc. The same way, JavaScript objects can have properties, which define their characteristics.

**Creating Objects in JavaScript:**

1. By object literal
2. By creating instance of Object directly (using new keyword)

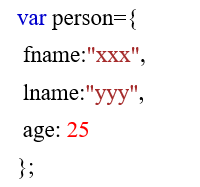
**By object literal:**

The syntax of creating object using object literal is given below:



Property and value is separated by colon(:).

**Example:**



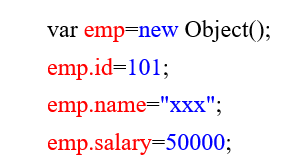
**By creating instance of Object directly (using new keyword):**

The syntax of creating object directly is given below:



Here, **new keyword** is used to create object.

**Example:**



**Accessing JavaScript Objects:**

The syntax for accessing the property of an object is:

*objectName.property*

or

*objectName*[“*property*”]

Accessing ‘fname’ from example 1 using dot operator,



Accessing ‘name’ form example 2 using [],

