Investigación de HTTP

Arquitectura Web

Profesor Antonio Cardeña

García Ramos, Bernardo

de Garay Hernández, Mauricio

Primavera 2021

Índice

Resumen

En este reporte se incluye toda la información necesaria para comprender HTTP sin necesidad de tener conocimientos previos del mismo. De igual manera, se profundiza un poco sobre ciertas funcionalidades y métodos de HTTP, al igual que de la versión 2.0 que actualmente es de las más utilizadas. Cabe mencionar que este reporte es más que nada una introducción a todo lo que es HTTP, por lo que somos breves y concisos en ciertas ideas y conceptos para no prolongar exageradamente el documento y la investigación.

Finalmente, en este reporte también incluímos conclusiones personales al final del documento con el propósito de entender la importancia de la investigación realizada. En aquella sección van a poder encontrar lo que aprendimos personalmente, así como ejemplos de cómo lo aprendido se podría ver reflejado en el ámbito laboral.

Contenido

¿Qué es HTTP?

“HTTP (o Protocolo de Transferencia de Hipertexto) es un protocolo de la capa de aplicación para la transmisión de documentos hipermedia, como HTML” (Mozilla, 2020). Con este protocolo, los clientes son capaces de realizar peticiones de datos y recursos a distintos servidores en la web.

Funcionalidad

De manera muy resumida vamos a explicar en pocos pasos que se necesita realizar para hacer una petición de datos y recursos del cliente al servidor:

1. El cliente (navegador) envía una petición de un documento HTML al servidor.
2. El servidor regresa la información necesaria para que el cliente pueda procesarla.
3. A continuación, el cliente envía más peticiones para solicitar scripts, hojas de estilo (CSS) y más datos como imágenes y videos.
4. El cliente une todos estos documentos y datos para poder formar la página web así como nosotros la visualizamos.

Métodos

- GET: Solicitar una representación de un recurso específico. Devuelve la representación en un formato en concreto.
- HEAD: Pedir una respuesta idéntica a la del GET pero sin el cuerpo.
- POST: Enviar una datos a un recurso en específico, posiblemente cambiando el estado o efectos secundarios en el servidor. Similar al GET pero un poco más específico.
- PUT: Actualiza o crea representaciones del recurso.
- DELETE: Elimina el recurso especificado.

- CONNECT: Inicia la comunicación en dos caminos con la fuente de los recursos solicitados.
- OPTIONS: Solicita las comunicaciones permitidas para un URL o servidor (Mozilla, 2020).
- TRACE: Crea un ciclo de mensajes de lazo cerrado hacia el *target*, funcionando como un gran mecanismo de depuración (Mozilla, 2020).
- PATCH: Aplica modificaciones parciales a un recurso.

HTTP: Previo a Versión 2

La primera versión de HTTP empezó como algo extremadamente sencillo. La petición consistía en una simple línea que empezaba con el único método que había (GET) seguido del camino del recurso, lo cual significaba que únicamente se podían transmitir archivos HTML. Un ejemplo de petición sería el siguiente:

GET /pagina.html

Poco a poco, HTTP fue evolucionando a HTTP 1.0. Se agregó una línea para el estado del código junto con la respuesta para que así el cliente supiera si había tenido éxito su petición. Se agregaron los HEADERS, permitiendo así que se pudieran transmitir metadatos u otro tipo de archivos y así hacer el protocolo más flexible.

Con la llegada de HTTP 1.1 se resolvieron muchos problemas que lo llevaron a ser un estándar como protocolo de comunicación cliente/servidor que hasta el día de hoy está presente. Con esta actualización las conexiones podían ser reutilizadas, se añadieron pipelines (o canales) para poder hacer una segunda petición entre una que aún no se terminaba de transmitir, también se añadieron headers para el host, y muchos otros cambios.

HTTP: Versión 2.0

La segunda versión de HTTP, o mejor conocida como HTTP/2, se volvió posiblemente la más popular/importante desde su llegada, ya que desde el 2016 más del 68% de todas las peticiones realizadas en la web usaban este mismo protocolo (Mozilla, 2020).

Una de las ventajas que proporciona esta nueva versión es que el protocolo es que es binario, por lo que puede ser mucho mejor optimizado que los de texto. De igual forma, este es un protocolo multiplexado, permitiendo así que se puedan realizar peticiones en paralelo en una misma conexión (baja latencia). Finalmente, también comprime los headers (encabezados) y permite llenar datos en el caché del cliente, lo cual lo lleva a minimizar sobrecarga del protocolo y hasta agregar compatibilidad.

Todas las ventajas anteriores al final lo llevan a ser un protocolo de baja latencia.

HTTP: Versión 3.0

El *Hypertext Transfer Protocol Version 3*, HTTP/3, se comenzó a desarrollar en 2018[1] y actualmente solamente está en algunas versiones de los navegadores más famosos, sin aún ser un estándar. Este nuevo protocolo tiene como meta el permitir una conexión mucho más rápida y segura que lo permitido por su predecesor (Bartlett, 2020).

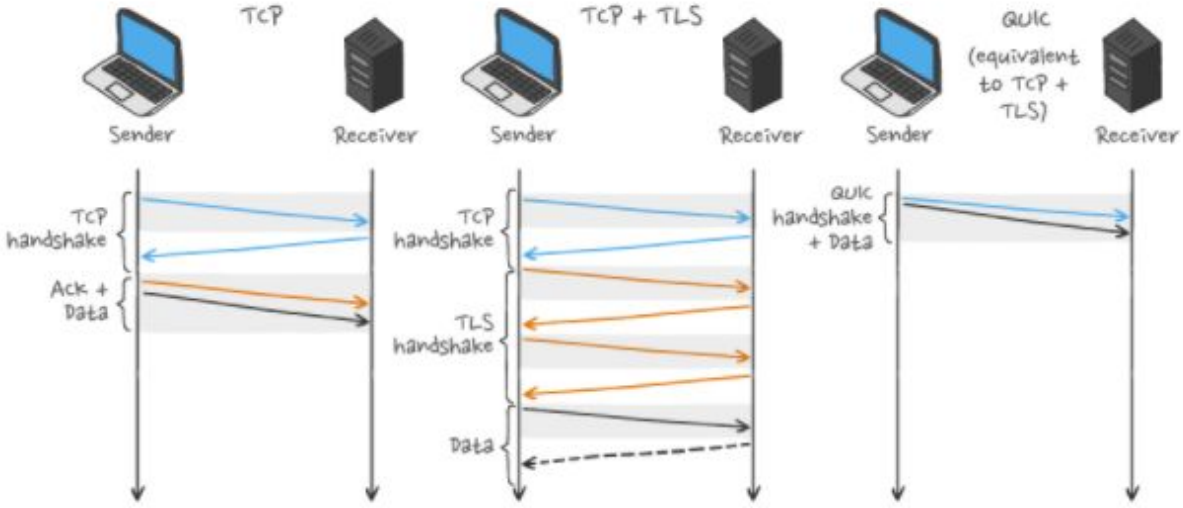
Un beneficio inmediato que traería este protocolo, es un mejor escalamiento y trabajo con aplicaciones en tiempo real, y para lograr esto se tienen que modificar varias propiedades que tenía HTTP/2. La más importante de estas, es reconocer la debilidad de la versión anterior en el ámbito de transporte (que lo hace utilizando

[1] A pesar de la gran adaptación de HTTP/2, se decidió desarrollar HTTP/3 por el gran crecimiento de usuarios en internet, y la enorme cantidad de plataformas adaptando las aplicaciones en tiempo real.

TCP), para así poder reemplazarlo con un protocolo conocido como QUIC. Este último permite la comunicación y transferencia multidireccional, a diferencia de la ordenada que tenían las versiones anteriores de HTTP (Bartlett, 2020).

Además de esto, cuenta con otros protocolos y estrategias para reducir la latencia entre comunicaciones mensajero-receptor. Un ejemplo de esto, es la modificación en cómo es que se comunican ambos miembros para reconocerse. A continuación, se puede ver en la figura 1 cómo es que esto difiere a las versiones anteriores, donde QUIC es el protocolo utilizado en HTTP/3 (Bartlett, 2020).

Figura 1. Comunicación mensajero/receptor en TCP vs QUIC. Disponible en: https://www.ably.io/topic/http3.



Claramente, todas estas ventajas (además de otras que también tiene) van a poder reducir en gran manera la latencia en las comunicaciones y transferencias, permitiendo así un mayor escalamiento en plataformas masivas y de tiempo real.

Conclusiones

Durante esta investigación, logramos aprender sobre HTTP, la gran importancia que tiene hoy en día en diversas aplicaciones, y por qué es tan importante conocer estos protocolos en nuestro campo. Además de esto, pensamos que un gran aprendizaje que quizás fue difícil de ver al principio, es sobre cómo es que estas tecnologías evolucionan tan rápido. Como se puede ver en la investigación, en 2016 HTTP/2 ya era un estándar, y hoy en día HTTP/3 ya está siendo adaptado[2]; esto nos dice mucho del hecho de que siempre debemos estar actualizados en los funcionamientos de todos estos protocolos si queremos ser exitosos en el campo de la programación. Es por eso que creemos que salimos de esta investigación con conocimientos muy importantes.

Además de esto, logramos pensar en muchísimas maneras que podríamos utilizar esto, hasta hoy en día. Más allá de los ejemplos sobre páginas web, uno de nosotros ahorita tiene un proyecto donde debe de enviar un archivo de un iphone a un servidor, y podría hacer esto utilizando el protocolo HTTP (que ahora conocemos a mayor medida gracias a este escrito). Hay un sinfín de aplicaciones más que utilizan el protocolo, tantas que pensamos que es obligatorio conocerlo a fondo si es que quieres ser competitivo en este campo.

[2] Pensamos que es especialmente impactante por la gran diferencia técnica que hay entre ambas versiones para el corto lapso de tiempo. ¡Es impactante!

Bibliografías y Referencias

Mozilla. (2020). “HTTP”. MDN Web Docs. Web. Recuperado el 30 de enero de 2020. https://developer.mozilla.org/es/docs/Web/HTTP.

Mozilla. (2020). “Evolution of HTTP”. MDN Web Docs. Web. Recuperado el 30 de enero de 2020. https://developer.mozilla.org/en-US/docs/Web/HTTP/Basics_of_HTTP/Evolution_of_HTTP.

Bartlett, J. (2020). “HTTP/3 Deep Dive”. Ably Realtime. Web. Recuperado el 1 de Febrero de 2020. https://www.ably.io/topic/http3.