https://doi.org/10.37815/rte.v35n3.1063
Artículos originales

Comparación del rendimiento en la transferencia de tráfico en servidores HTTP/2 y QUIC

Comparison of traffic transfer performance on HTTP/2 and QUIC servers

Jairo Valle[1] https://orcid.org/0009-0003-6808-0131, Rommel Torres[1] https://orcid.org/0000-0003-2313-0118, Liliana Enciso[1] https://orcid.org/0000-0002-2918-9033, Patricia Ludeña[1] https://orcid.org/0000-0002-8909-4837

[1]*Universidad Técnica Particular de Loja*, Loja, Ecuador
jsvalle1@utpl.edu.ec, rovitor@utpl.edu.ec, lenciso@utpl.edu.ec, pjludena@utpl.edu.ec

Enviado: 2023/07/16
Aceptado: 2023/09/19
Publicado: 2023/12/30

Resumen

Desde el surgimiento de la World Wide Web en los 90´s, el protocolo HTTP (Hypertext Transfer Protocol) ha permitido el intercambio de datos entre cliente y servidor, ante el constante incremento de las comunicaciones de red donde la transferencia de datos es cada vez más rápida, segura y fiable; este protocolo ha ido evolucionando en diferentes versiones hasta implementar al día de hoy la versión denominada QUIC (Quick UDP Internet Connections).

En el presente trabajo se evaluó mediante el uso de los servidores web dedicados OpenLiteSpeed y Nginx, así como el servidor comercial Hostinguer, el rendimiento en la transferencia de tráfico normal y multimedia mediante el protocolo HTTP en sus versiones HTTP/2 y QUIC.

Para esto se configuró un entorno web que permitiera establecer la comunicación y transferencia de archivos bajo la arquitectura cliente-servidor. En los servidores utilizados, se implementó una página web desarrollada en WordPress que posee la capacidad de cargar archivos de diversos formatos hacia el servidor.

Finalmente se establecieron 4 escenarios de pruebas para comparar de forma práctica el proceso de comunicación y transferencia entre cliente y servidor, apoyados de la herramienta Wireshark, la cual nos permitió monitorear y gestionar el tráfico de red.

Palabras clave: protocolo, transferencia, versiones, escenarios, tráfico, monitorear.

Como citar: Valle, J., Torres, R., Enciso, L. & Ludeña, P. (2023). Comparación del rendimiento en la transferencia de tráfico en servidores HTTP/2 y QUIC. *Revista Tecnológica - Espol, 35(3), 68-82.* http://www.rte.espol.edu.ec/index.php/tecnologica/article/view/1063

Abstract

Since the emergence of the World Wide Web in the 90s, The HTTP (Hypertext Transfer Protocol) has enabled data exchange between customers and servers. Considering the constant increase in network communications where data transfer is becoming faster, safer, and more reliable, this protocol has evolved through different versions, culminating in the current implementation known as QUIC (Quick UDP Internet Connections).

In this study, we evaluated through the use of the dedicated web servers Open LiteSpeed and Nginx, as well as the commercial server Hostinguer, the performance in normal traffic transfer and multimedia through the HTTP protocol in its versions: HTTP/2 and QUIC.

This was achieved by configuring a web environment to establish communication and file transfer under the customer-server architecture. On the employed servers, a web page developed in WordPress was deployed, which possessed the capability to upload files of various formats to the server.

Finally, four test scenarios were established to compare, in a practical way, the communication and transfer process between the customer and server. The monitoring and management of network traffic were facilitated by the Wireshark tool.

Keywords: protocol, transfer, versions, scenarios, traffic, monitor.

Introducción

El protocolo HTTP (Hypertext Transfer Protocol) se usa en los navegadores y servidores desde 1991 y ha sido uno de los protocolos de comunicación más utilizados en Internet (Kyaw, 2019). El protocolo HTTP utiliza un esquema de transacciones basado en solicitud/respuesta (Murthy et al., 2023). La conexión se establece cuando un cliente envía una petición mediante un mensaje hacía el servidor y este responde con un mensaje de igual característica, detallando la operación y su resultado. Todas las operaciones pueden adjuntar un objeto o recurso sobre el que actúan, cada objeto web es conocido por su URL (Henríquez, 2017). Desde su implementación la mayoría del tráfico que circula en Internet es enviado mediante el uso de este protocolo, además, en los últimos años ha aumentado el número de aplicaciones web que lo utilizan en diferentes recursos como son imágenes, CSS, JavaScript, etc., así como el aumento considerable de conexiones seguras dentro de la red (Vega, 2014).

El protocolo HTTP es considerado el protocolo más utilizado a nivel de la capa de aplicación, siendo el protocolo fundamental para el acceso de los datos en Internet (Oliveira, 2020).

Desde la aparición de la World Wide Web, desde 1991, el protocolo HTTP en su primera versión denominada HTTP/0.9 permitía el intercambio de datos por la web sin procesar su información, con la aparición de HTTP/1.0 en 1996 (Tomás, 2021) se mejoró el protocolo incorporando metainformación sobre los datos y una semántica de solicitud-respuesta, sin embargo, esta versión no toma en cuenta el diferenciar una jerarquía de proxys, almacenamiento de caché o conexiones persistentes (Fielding et al., 1999), es por ello que surge en 1997 HTTP/1.1 el cual se mantiene hasta 2015 donde surge HTTP/2 (Tomás, 2021) como solución a la petición múltiple de conexiones al servidor que la versión HTTP/1.0 solicitaba para la comunicación con el cliente para reducir la latencia, es por ello, que HTTP/2 adiciona el protocolo de seguridad TLS (Transport Layer Security). Google Chrome es uno de los navegadores más utilizados por los usuarios en Internet, en 2022 se calculó que este navegador es utilizado por los usuarios de Internet en un 68.56% (NetApplications.com, 2017), por lo que se puede destacar el liderazgo de la empresa Google en el servicio de navegadores web y por ende una incidencia significativa del protocolo QUIC entre sus clientes y servidores (Espinosa, 2019).

TLS se define como un protocolo orientado a la seguridad en la capa de transporte de manera criptográfica, proporciona encriptación y autenticación de todas las partes en la comunicación de la red. Publicado en 1999 por la RFC2246 (Rueda, 2019), se constituye en el sucesor del protocolo SSL (Secure Sockets Layer) y hasta la actualidad se ha publicado 4 versiones de este, siendo la versión TLS 1.3 publicado en 2018 su último borrador con el que se cuenta (Priego, 2018). Publicado en agosto de 2018 en la RFC8446, TLS 1.3 proporcionó una serie de actualizaciones de parámetros de seguridad y mejorar el rendimiento en la comunicación (Romero, 2020), tiene como objetivo principal el ofrecer una canal fiable y seguro para la comunicación entre pares de la red bajo un canal seguro y ordenado, para tal efecto debe poseer características de seguridad como criptografía asimétrica en la autenticación, confidencialidad de la información dado que sólo los puntos finales de la comunicación tienen capacidad de visualizar el contenido e integridad de los datos de forma que no pueden ser modificados (Rescorla, 2018).

QUIC (Quick UDP Internet Connections) es un protocolo que nace de la necesidad de reducir la latencia que genera TCP al momento de establecer conexión (Albasrawi, 2020), además de implementar nuevas características y mejoras en cuanto a la seguridad en la transmisión (Fernández et al., 2021).

QUIC es un protocolo situado en la capa de transporte que funciona bajo el User Datagram Protocol (UDP), y desarrollado por Google desde 2012, actualmente se encuentra en discusión por el equipo de desarrollo de Google y el Internet Engineering Task Force (IETF) por lo que se cuenta con 2 variaciones del mismo, QUIC de Google (GQUIC) y IETF QUIC (IQUIC) (Thomas et al., 2019).

Las principales diferencias entre QUIC y TCP se presentan en la Tabla 1.

Tabla 1

Comparación entre QUIC Y TCP

CRITERIO	TCP	QUIC
Conexión	Para establecer conexión TCP utiliza unprotocolo de enlace de 3 vías.	Para la conexión QUIC utiliza un paquetede inicio.
Seguridad	Utiliza TLS que consiste en cifrado de datos.	Utiliza encriptado y autenticación deextremo a extremo.
Paquetes	TCP se encarga de enviar y recibir paquetes del servidor.	QUIC sólo envía paquetes al servidor loque mejora la latencia.

Para establecer la conexión en QUIC el cliente primeramente usa un mensaje de un paquete para obtener información del servidor denominado como Handshake inicial y así completar el protocolo de enlace. El cliente envía un mensaje Inchoate CHLO al servidor, este es un Client Hello incompleto y el servidor le responde con un mensaje Reject, el cual contiene el token de dirección de origen, una firma digital, una clave privada y los certificados del servidor que van a ayudar al cliente a continuar con la comunicación con el cliente (Espinosa, 2019).

Existen diversos trabajos orientados a la comparación de las diferentes versiones del protocolo HTTP y QUIC entre los que podemos destacar: “Análisis de velocidad de acceso a sitios web comparando protocolo TCP tradicional con SSL vs protocolo QUIC” (Iglesias & Guaman, 2021), en dónde se mide los tiempos de carga a sitios en la web de manera especial los ofrecidos por Google que soportan tanto el protocolo QUIC y TCP; así mismo en

"Evaluación del uso del protocolo QUIC en Internet" (Espinosa, 2019), se hace uso de los servidores Apache y Cloudflare para la navegación en sitios web que permiten QUIC de la versión IETF. Luego del análisis realizado a diferentes trabajos en los que se utiliza las versiones HTTP y QUIC, se puede destacar que el presente trabajo se orienta a establecer diferentes escenarios de pruebas rigurosos hacia los servidores propuestos, con el fin de generar pruebas más exhaustivas y equitativas de condiciones, mediante la carga de archivos en diferentes formatos (TXT, JPG y MP4) y así poder establecer comparaciones a nivel de protocolos, servidores, limitaciones de red y formatos utilizados.

El objetivo de la presente investigación se centra en analizar el rendimiento en la transferencia de tráfico normal y multimedia mediante la implementación de servidores que trabajen con los protocolos HTTP y QUIC con el fin de establecer sus diferencias y particularidades mediante el establecimiento de diversos escenarios de prueba.

El presente trabajo está organizado de la siguiente manera:

En la sección de Materiales y Métodos se establece la arquitectura a utilizar, así como las herramientas de software y los servidores seleccionados, así como también los sitios web utilizados en cada servidor con sus respectivos dominios; luego se procede a establecer diferentes escenarios de pruebas con sus características principales.

La sección de Resultados y Discusión se evidencia los resultados obtenidos en el desarrollo de cada escenario de prueba propuesto, y se genera el detalle y análisis en la comparación de características relevantes encontradas en cada uno de ellos, así como la inclusión de gráficas estadísticas que aportan a la discusión y comparación de forma gráfica.

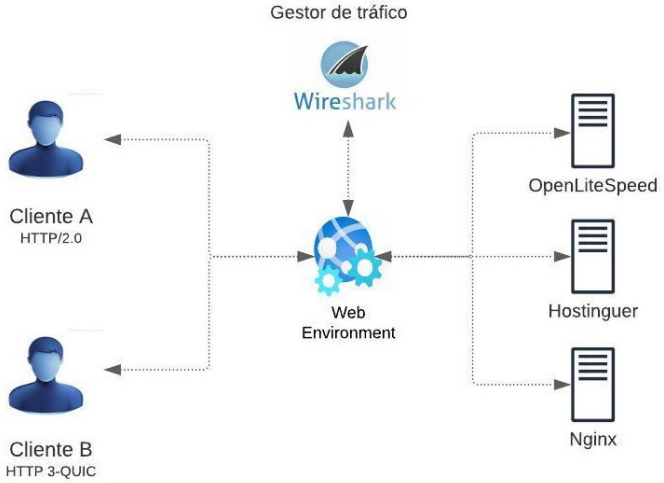
Finalmente, en la sección de conclusiones se genera a partir del estudio y discusión de los resultados obtenidos, las principales características encontradas en la comparación de los protocolos HTTP y QUIC.

Materiales y Métodos

Una vez analizados los fundamentos y características más relevantes de las versiones a través del tiempo sobre el protocolo HTTP, se procede a la planeación para configurar un entorno web que permita establecer 4 escenarios de prueba sobre 3 servidores web.

Figura 1

Arquitectura del entorno web



En la Figura 1, se muestra el entorno web establecido con la presencia de 2 host o clientes con capacidades de comunicación mediante protocolo HTTP/2 y QUIC, así mismo se establece un total de 3 servidores web; 2 servidores web dedicados (OpenLiteSpeed y Nginx) y 1 servidor comercial (Hostinguer).

Finalmente, la presencia de un gestor de tráfico (Wireshark) que se trata de una herramienta de captura de paquetes del tráfico de red, con la finalidad de realizar un análisis y medición del uso de los diferentes protocolos en la transferencia de archivos multimedia y de texto.

Componentes de software

Los componentes de software a utilizar para la realización del entorno web y comunicación cliente – servidor se describen a continuación en la Tabla 2, donde se identifica la herramienta a utilizar y una breve descripción de la misma.

Tabla 2

Componentes a utilizar para la generación del entorno web

Componente	Descripción
Digital Ocean	Servicio de alojamiento cloud privado, dónde se permite la creación de droplets, el cual permite configurar un servidor remoto en un sistema operativo específico (Morocho Troya, 2022).
Droplet	Consiste en una máquina virtual que opera como un servidor virtual privado (VPS) y con capacidad de ser multipropósito (Rodríguez, 2020).
Namecheap	Servicio de selección y registro de dominios web a bajos costos, es el segundo registrador de dominios más usado en el mundo, los usuarios pueden adquirir y vender sus dominios, así como posee una gran facilidad en las configuraciones DNS (*Domain Name System*) del dominio (Ke et al., 2023).
Wordpress	Es un sistema de gestión de contenidos (CMS) enfocado en la creación y gestión de aplicaciones y páginas web dinámicas (Calle-González, 2020).
Wireshark	Herramienta sniffer o analizador de red de código abierto, el cual nos permite capturar los paquetes de datos del tráfico que se genera en una red para su análisis, posee una interfaz gráfica, generación de reportes y aplicación de filtros que nos permite identificar protocolos, puertos, direcciones ip y más(Bock, 2022) .
Power Bi	Es una herramienta de análisis de datos con la capacidad de filtrar y generar reportes gráficos interactivos a través de los datos obtenidos (Bermeo-Pérez & Campoverde-Molina, 2020).

Servidores

Para el desarrollo del entorno web que permite la carga de archivos, monitoreo de la transferencia cliente – servidor y posteriormente la comparación de resultados, se opta por el uso y configuración de dos servidores dedicados y uno comercial, los cuales tienen la capacidad de comunicación y transferencia mediante protocolo HTTP/2 y QUIC.

Servidor OpenLiteSpeed

Este servidor web es la versión gratuita de LiteSpeed Technologies, entre sus

principales características se puede destacar su notable velocidad en comparación al servidor web Apache, así como su alto rendimiento, estabilidad y eficiencia (Sandra, 2022). Para el desarrollo de este trabajo, se configura OpenLiteSpeed en el sistema operativo Ubuntu 22.04 mediante un ambiente cloud con Digital Ocean, como punto de partida se debe generar un nuevo droplet en la plataforma bajo la virtualización del sistema operativo Ubuntu 22.04, una vez instalado el droplet obtendremos una IP pública la cuál podremos configurar en el DNS del dominio que previamente hemos obtenido en Namecheap. En este servidor el dominio a configurar será: www.pruebaquic.online.

Servidor Hostinguer

Hostinger es un proveedor de alojamiento y dominios web privado, entre sus características podemos destacar que es una plataforma ágil y fácil de configurar en la creación de un nuevo dominio y página web, por lo cual se obtiene un plan de hosting y se configura el dominio www.pruebaquic2.online, luego de esto se crea un sitio web mediante la gestión de contenidos en Wordpress.

Servidor Nginx

El servidor Nginx sale a la luz en 2004 y su creador es Igor Sysoev, utilizado por grandes sitios web como WordPress, Hulu y MochiMedia, se caracteriza por su estabilidad, seguridad y su fácil configuración demostrando una gran eficiencia (Reese, 2008). Para el presente trabajo, este servidor se configura mediante la virtualización de un servidor LEMP (Linux, Nginx, MySQL, PHP) en la plataforma de Digital Ocean, una vez instalado el droplet obtendremos una Ip pública la cuál podremos configurar en el DNS del dominio www.pruebaquic1.online mediante Namecheap, se puede verificar que el servidor se ha instalado correctamente si al ingresar al dominio se presenta el mensaje de bienvenida.

Interfaz de usuario

Una vez realizada la configuración de cada uno de los servidores a utilizar, se realiza la instalación y configuración de WordPress, así como el diseño de una página sencilla que posee un formulario para la carga de los archivos. Esta página se replica en los 3 servidores para así poseer características similares en cada uno de los escenarios de prueba.

La interfaz de nuestra página realizada en WordPress posee un formulario de rápido acceso para realizar la operación de selección y carga de un archivo, proceso por el cual se da la capacidad al cliente de poder elegir un archivo almacenado en su equipo y hacer la carga de este hacia el servidor previamente configurado.

Como se ha mencionado, esta interfaz se encuentra disponible en los 3 servidores configurados mediante los siguientes dominios públicos detallados en la Tabla 3.

Tabla 3

Ip pública y dominio utilizado en cada servidor

SERVIDOR	IP PÚBLICA	DOMINIO
OpenLiteSpeed	159.223.128.63	pruebaquic.online
Hostinguer	89.117.139.207	pruebaquic2.online
Nginx	142.93.253.6	pruebaquic1.online

Escenarios de prueba

Una vez establecido el entorno web se plantean los escenarios de prueba para evaluar la transferencia de tráfico normal y multimedia mediante los protocolos HTTP/2 y QUIC. Por tanto, se describen a continuación los escenarios a realizar.

Escenario 1

En este escenario se limita el ancho de banda de la red en 500Kbps, y serealiza en los servidores OpenLiteSpeed, Hostinguer y Nginx la carga de los archivos que se detallan en la Tabla 4, primero mediante el protocolo HTTP/2 y luego mediante QUIC.

Tabla 4

Archivos a utilizar en Escenario 1

TIPO DE ARCHIVO	TAMAÑO (Kb)
JPG	3030322
MP4	3040771
TXT	3031970

Escenario 2

En este escenario se limita el ancho de banda de la red en 1000 Kbps, asícomo la generación de latencia en la transferencia configurado en 10ms, se realiza en los servidores OpenLiteSpeed, Hostinguer y Nginx la carga de los archivos que se detallan en la Tabla 5, primero mediante el protocolo HTTP/2 y luego mediante QUIC.

Tabla 5

Archivos a utilizar en Escenario 2

TIPO DE ARCHIVO	TAMAÑO (Kb)
JPG	5025883
MP4	5038033
TXT	5030716

Escenario 3

En este escenario se limita el ancho de banda de la red en 4000 Kbps, asícomo la generación de latencia en la transferencia configurado en 15ms y por último la pérdida de paquetes en un 4%, se realiza esta prueba en los servidores OpenLiteSpeed y Hostinguer la carga de los archivos que se detallan en la Tabla 6, primero mediante el protocolo HTTP/2 y luego mediante QUIC.

Tabla 6

Archivos a utilizar en Escenario 3

TIPO DE ARCHIVO	TAMAÑO (Kb)
JPG	8028577
MP4	8042005
TXT	8025669

Resultados y Discusión

En este apartado se muestran los resultados obtenidos a partir de los escenarios de prueba descritos anteriormente.

Resultado Escenario 1

En el escenario 1 con los parámetros establecidos en la sección anterior, mediante la Tabla 7 se especifica el servidor utilizado, así como el ancho de banda correspondiente a este escenario definido en 500Kbps, bajo que protocolo se realiza la prueba, el tipo de archivo y el resultado de las métricas de evaluación obtenidas desde la herramienta Wireshark en cada sesión de prueba.

Tabla 7

Resultados obtenidos Escenario 1

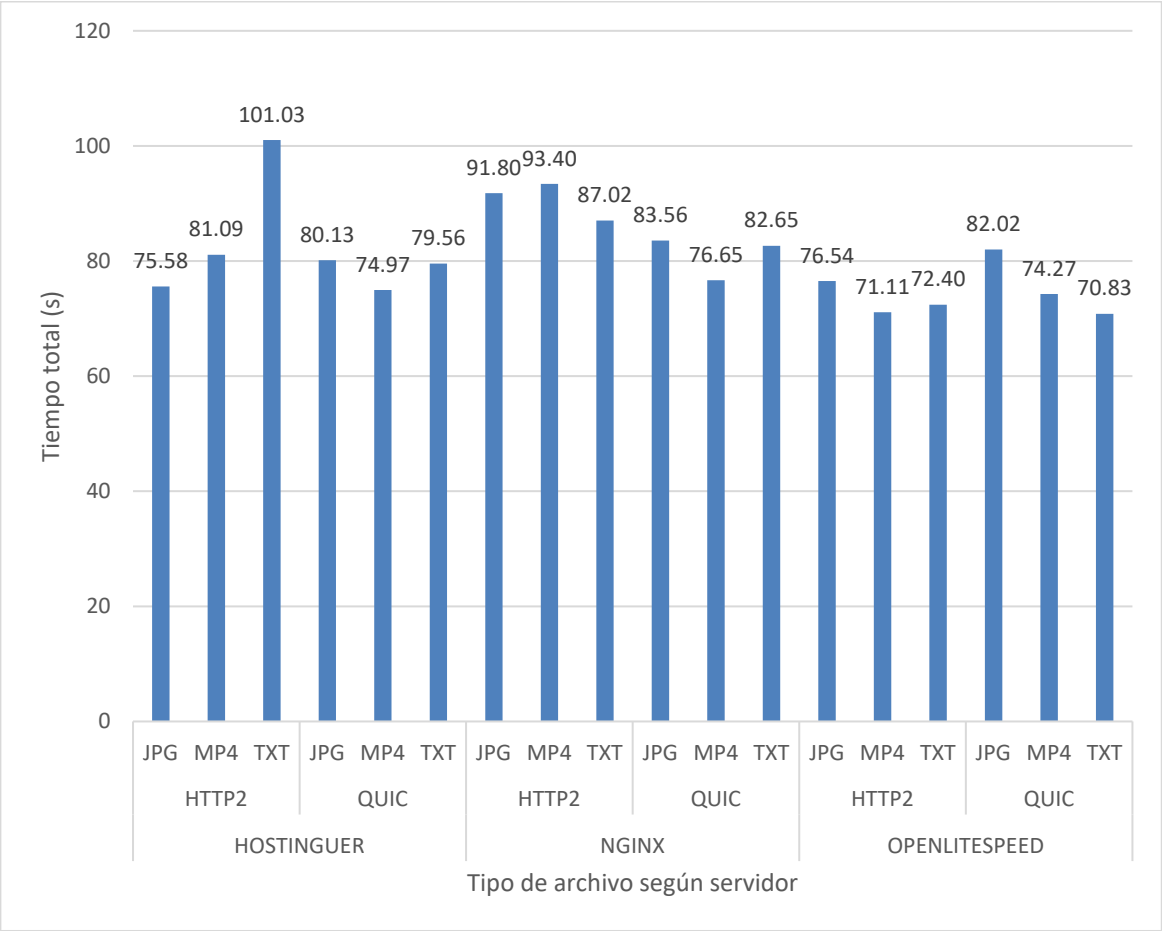
SERVIDOR	ANCHO DE BANDA (kbps)	PROTOCOLO	ARCHIVO	TAMAÑO (Mb)	RTT (ms)	PAQUETES CAPTURADOS	TIEMPO (s)	PESO (Kb)
OPEN LITE SPEED	500	QUIC	JPG	3	0.0055	5385	82.01	4833
OPEN LITE SPEED	500	QUIC	MP4	3	0.0054	5510	74.27	4988
OPEN LITE SPEED	500	QUIC	TXT	3	0.0041	5177	70.82	4640
HOSTINGUER	500	QUIC	JPG	3	0.0005	5494	80.12	4873
HOSTINGUER	500	QUIC	MP4	3	0.0097	5716	74.97	5121
HOSTINGUER	500	QUIC	TXT	3	0.0006	5458	79.55	4793
NGINX	500	QUIC	JPG	3	0.0042	6152	83.56	4901
NGINX	500	QUIC	MP4	3	0.0025	6586	76.65	5255
NGINX	500	QUIC	TXT	3	0.0036	6103	82.65	4896
OPEN LITE SPEED	500	HTTP2	JPG	3	0.06	4580	76.53	4324
OPEN LITE SPEED	500	HTTP2	MP4	3	0.08	4848	71.10	4331
OPEN LITE SPEED	500	HTTP2	TXT	3	0.0793	4941	72.40	4444
HOSTINGUER	500	HTTP2	JPG	3	0.1066	5188	75.58	4428
HOSTINGUER	500	HTTP2	MP4	3	0.1151	4961	81.08	4317
HOSTINGUER	500	HTTP2	TXT	3	0.1129	6677	101.02	6018
NGINX	500	HTTP2	JPG	3	0.0873	6808	91.80	5672
NGINX	500	HTTP2	MP4	3	0.0849	6941	93.40	5731
NGINX	500	HTTP2	TXT	3	0.0798	6266	87.02	5107

En la Figura 2 se aprecia una comparación del tiempo total de carga entre los protocolos HTTP/2 y QUIC, se puede concluir que, en un ancho de banda de 500 Kbps, los servidores Hostinguer y Nginx bajo el protocolo QUIC obtuvo menor tiempo de carga en comparación a HTTP/2, en cambio en el servidor OpenLiteSpeed la diferencia entre los dos protocolos es mínima provocando que HTTP/2 obtenga un menor tiempo de carga de 7 segundos menos que QUIC. Lo que nos indica que el protocolo QUIC obtiene menor tiempo de carga en 2 de los 3

servidores puestos a prueba.

Figura 2

Tiempo total de carga según protocolos HTTP/2 y QUIC en Escenario 1



Resultado Escenario 2

Los resultados obtenidos en el escenario 2 se detallan en la Tabla 8, dónde se especifica el servidor utilizado, así como el ancho de banda correspondiente a este escenario definido en 1 Mbps, bajo que protocolo se realiza la prueba, el tipo de archivo y el resultado de las métricas de evaluación obtenidas desde la herramienta Wireshark en cada sesión de prueba.

Tabla 8

Resultados obtenidos Escenario 2

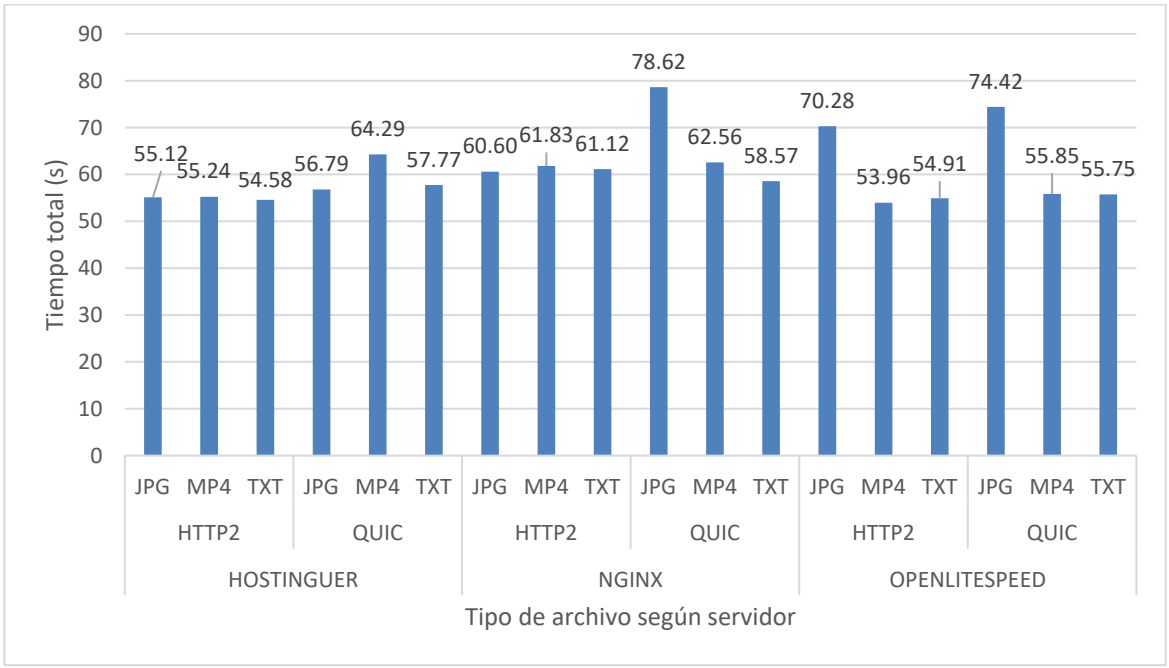
SERVIDOR	ANCHO DE BANDA (Mbps)	PROTOCOLO	ARCHIVO	TAMAÑO (Mb)	RTT (ms)	PAQUETES CAPTURADOS	TIEMPO (s)	PESO (kb)
OPEN LITE SPEED	1	QUIC	JPG	5	0.0134	7656	74.41	6981
OPEN LITE SPEED	1	QUIC	MP4	5	0.0001	7770	55.85	7083
OPEN LITE SPEED	1	QUIC	TXT	5	0.0003	8063	55.74	7394
HOSTINGUER	1	QUIC	JPG	5	0.0001	7662	56.78	6938
HOSTINGUER	1	QUIC	MP4	5	0.0055	7777	64.28	7059

SERVIDOR	ANCHO DE BANDA (Mbps)	PROTOCOLO	ARCHIVO	TAMAÑO (Mb)	RTT (ms)	PAQUETES CAPTURADOS	TIEMPO (s)	PESO (kb)
HOSTINGUER	1	QUIC	TXT	5	0.0021	7876	57.77	7154
NGINX	1	QUIC	JPG	5	0.0145	7675	78.62	6992
NGINX	1	QUIC	MP4	5	0.0042	7798	62.56	7156
NGINX	1	QUIC	TXT	5	0.0005	7956	58.56	7056
OPEN LITE SPEED	1	HTTP2	JPG	5	0.0916	6835	70.27	6286
OPEN LITE SPEED	1	HTTP2	MP4	5	0.0903	6998	53.95	6444
OPEN LITE SPEED	1	HTTP2	TXT	5	0.0948	6947	54.90	6467
HOSTINGUER	1	HTTP2	JPG	5	0.1083	7327	55.12	6758
HOSTINGUER	1	HTTP2	MP4	5	0.1074	7020	55.24	6491
HOSTINGUER	1	HTTP2	TXT	5	0.1028	7860	54.57	6927
NGINX	1	HTTP2	JPG	5	0.0828	8215	60.60	7228
NGINX	1	HTTP2	MP4	5	0.0957	8053	61.82	7189
NGINX	1	HTTP2	TXT	5	0.0365	7841	61.12	7246

En la Figura 3, se muestra una comparación de tiempo y paquetes capturados según el formato de archivo en cada uno de los servidores puestos a prueba, en donde podemos destacar que en el servidor OpenLiteSpeed, el número de paquetes capturados bajo el protocolo HTTP/2 es significativamente más bajo que QUIC y la transferencia se realiza en menor tiempo, es así que podemos considerar que en este servidor tanto HTTP/2 como QUIC la transferencia del archivo de formato JPG es quien más tardó a diferencia de los demás archivos.

Figura 3

Tiempo total de transferencia según formato de archivo en Escenario 2



En el servidor Hostinguer sucede lo mismo, el número de paquetes capturados es más bajo en el protocolo HTTP/2 que en el protocolo QUIC. En el servidor Nginx el tiempo de transferencia es menor en HTTP/2 que en el protocolo QUIC, pero el número de paquetes es mayor en el protocolo HTTP/2 que en el protocolo QUIC; lo que nos indica que el protocolo HTTP/2 obtiene mejores resultados en un ambiente donde el ancho de banda es de 1 Mbps, con una latencia de 10 m.s y con archivos de tamaño de 5Mb.

Resultado Escenario 3

Los resultados obtenidos en el escenario 3 se detallan en la Tabla 9, dónde se especifica el servidor utilizado, así como el ancho de banda correspondiente a este escenario definido en 4 Mbps, bajo que protocolo se realizó la prueba, el tipo de archivo y el resultado de las métricas de evaluación obtenidas desde la herramienta Wireshark en cada sesión de prueba.

Tabla 9

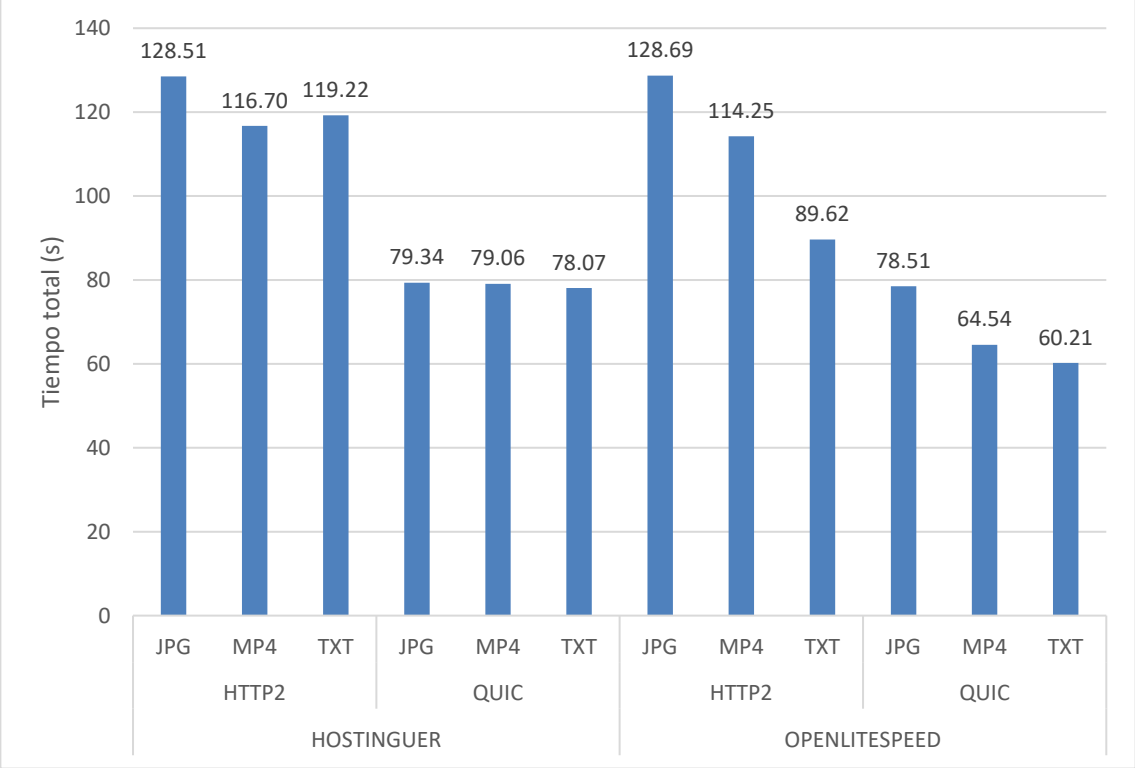
Resultados obtenidos Escenario 3

SERVIDOR	ANCHO DE BANDA (Mbps)	PROTOCOLO	ARCHIVO	TAMAÑO (Mb)	RTT (ms)	PAQUETES CAPTURADOS	TIEMPO (s)	PESO (kb)
OPEN LITE SPEED	4	QUIC	JPG	5	0	10449	78.50	9273
OPEN LITE SPEED	4	QUIC	MP4	5	0	10397	64.54	9278
OPEN LITE SPEED	4	QUIC	TXT	5	0	10339	60.21	9253
HOSTINGUER	4	QUIC	JPG	5	0.0039	10638	79.34	9299
HOSTINGUER	4	QUIC	MP4	5	0	10675	79.05	9311
HOSTINGUER	4	QUIC	TXT	5	0	10688	78.06	9338
OPEN LITE SPEED	4	HTTP2	JPG	5	0.0016	9361	128.69	9130
OPEN LITE SPEED	4	HTTP2	MP4	5	0.0012	9409	114.25	9154
OPEN LITE SPEED	4	HTTP2	TXT	5	0.0016	9254	89.62	9129
HOSTINGUER	4	HTTP2	JPG	5	0.0045	9162	128.50	9140
HOSTINGUER	4	HTTP2	MP4	5	0.0665	10990	212.85	10
HOSTINGUER	4	HTTP2	TXT	5	0.0881	10958	228.78	10

En este escenario cabe destacar que se realizó una serie de pruebas de carga a los servidores mediante el protocolo HTTP/2 con diferentes porcentajes en la pérdida de paquetes, al ser un escenario crítico con la presencia de latencia, pérdida de paquetes y con un ancho de banda limitado a 4Mbps se pudo evidenciar que no se completaba el proceso de transferencia con configuraciones superiores al 4% de pérdida de paquetes. Es por esto que se realizó las pruebas con los 2 servidores que presentan mejores resultados en los escenarios anteriores que son OpenLiteSpeed y Hostinguer.

Figura 4

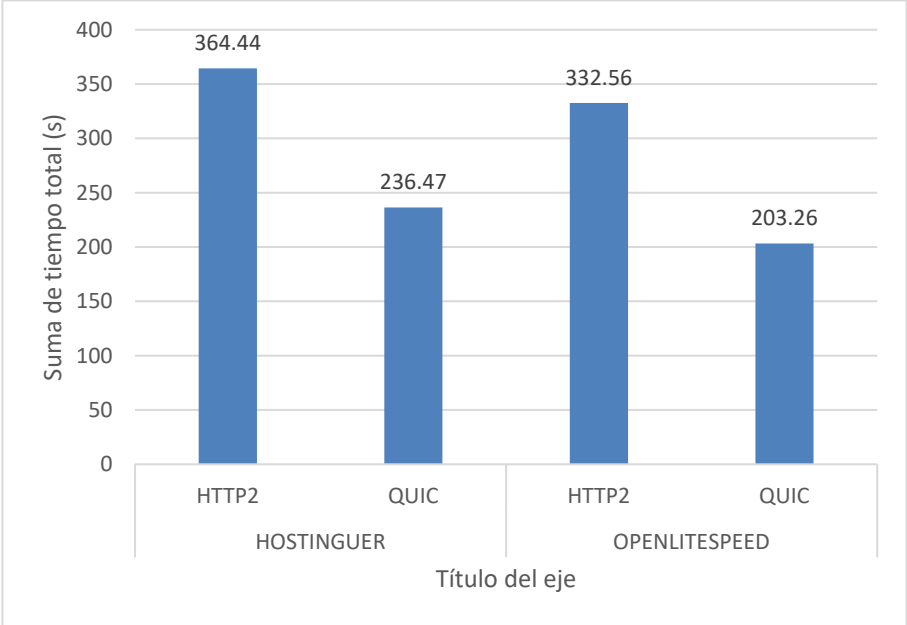
Tiempo total de transferencia según formato de archivo en Escenario 3



Como se puede apreciar en la Figura 4, en este escenario el protocolo QUIC obtiene menor tiempo total en la transferencia de archivos desde cliente hacia el servidor en cada uno de los formatos utilizados, siendo más notoria la diferencia existente en el formato de archivo TXT. El servidor OpenLiteSpeed es quien obtiene los mejores tiempos de transferencia en comparación a Hostinguer lo cual podemos apreciar a mayor detalle en la Figura 5.

Figura 5

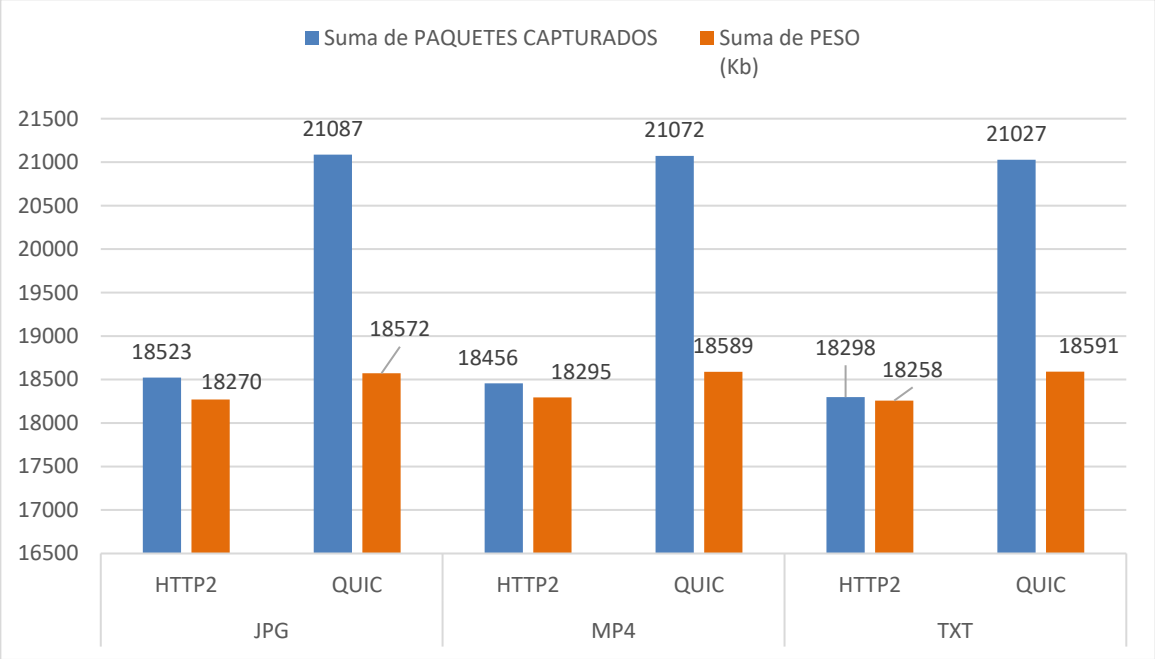
Tiempo total de transferencia según protocolo de cada servidor en Escenario 3



Por último, se realiza una comparación respecto al número de paquetes capturados y el peso total de la transferencia realizada con cada uno de los formatos de archivo utilizados en este escenario. Ver Figura 6.

Figura 6

Comparación de total de paquetes capturados y peso total según formato de archivo en Escenario 3



QUIC genera una mayor cantidad de paquetes con respecto a HTTP/2 y esto implica que el peso total de transferencia será mayor, se puede atribuir este resultado a que QUIC dado que su transmisión se basa en UDP genera datagramas de menor longitud, por ende, será mayor el número de paquetes generados y más peso ya que se debe tomar en cuenta los bytes extras que se generan de las cabeceras de cada datagrama.

Conclusiones

El trabajo realizado para comparar el rendimiento en servidores web tiene la intención de poder evidenciar las diferencias existentes en cuanto a la transferencia de tráfico normal y multimedia que se tiene presente en la navegación por Internet hoy en día. La versión emergente del protocolo HTTP denominado QUIC presenta, bajo los escenarios desarrollados en este trabajo, características importantes referente a su desempeño como son:

En el escenario 1 dónde se limita el ancho de banda a 500 Kbps, la sumatoria del tiempo total de carga de todos los servidores utilizados bajo el protocolo HTTP/2 es de 749.97 segundos, mientras que mediante el protocolo QUIC es de 704.64 segundos, dando una diferencia de 45.33 segundos, lo que define al protocolo QUIC como el más óptimo en la transferencia de los archivos en estas condiciones de prueba dado que genera menos tiempo de transmisión.

En el escenario 2 configurado con un ancho de banda de 1 Mbps y latencia de 10 m.s, el archivo con formato JPG mediante el protocolo de transferencia QUIC presenta una diferencia considerable en relación a los otros 2 formatos utilizados (TXT, MP4), dando un

tiempo total excedente de 10 segundos, aun cuando generó 100 paquetes de captura menos que los otros formatos, mientras que haciendo la comparación frente al protocolo HTTP/2 se puede notar una diferencia de 40 segundos más de tiempo y un generación de 1295 paquetes capturados de diferencia, se puede concluir que bajo estas condiciones de prueba el formato JPG en el protocolo QUIC demuestra que tiende a demorar de manera significativa frente a los otros formatos y más aún frente al protocolo HTTP/2.

En el escenario 3 donde se limita el ancho de banda a 4 Mbps, latencia de 15 m.s y pérdida de paquetes al 4%, la sumatoria de los servidores bajo el protocolo QUIC obtuvo un tiempo total de 439.72 segundos, mientras que HTTP/2 obtuvo un tiempo total de 696.99 segundo, generando una diferencia de 257.27 segundos, es así que podemos concluir que el protocolo QUIC es más eficiente en la transferencia de archivos con entornos de red con un ancho de red limitado, con presencia de latencia y pérdida de paquetes, ya que realiza la transferencia en un tiempo menor que HTTP/2

En un entorno de red dónde se simula pérdida de paquetes mayor a 6% y con latencia mayor a 15 m.s, podemos destacar que el protocolo HTTP/2 en los diferentes servidores utilizados, obtuvo una capacidad de respuesta muy deficiente, no completando la fase de subida de los archivos en 2 servidores (Nginx y Hostinguer) y generando un error en la transferencia, no obstante, al hacer uso del protocolo QUIC bajo los mismos parámetros de simulación; generó una respuesta positiva al primer intento en 2 servidores (OpenLiteSpeed y Hostinguer), completando de manera exitosa el proceso de subida y transferencia de los archivos, por ende podemos concluir que el protocolo QUIC responde de mejor manera a entornos de red limitados y críticos por la presencia de latencia y pérdida de paquetes.

Referencias

Albasrawi, S. T. (2020). Performance analysis of Google's Quick UDP Internet Connection Protocol under Software Simulator. *Journal of Physics: Conference Series*, 1591(1), 012026.

Bermeo-Pérez, S. K., & Campoverde-Molina, M. A. (2020). Implementación de inteligencia de negocios, en el inventario de la Cooperativa GranSol, con la herramienta Power BI. *Revista Científica FIPCAEC* (Fomento de La Investigación y Publicación Científico-Técnica Multidisciplinaria). ISSN: 2588-090X. Polo de Capacitación, Investigación y Publicación (POCAIP), 5(16), 240–266.

Bock, L. (2022). Learn Wireshark: A Definitive Guide to Expertly Analyzing Protocols and Troubleshooting Networks Using Wireshark. *Packt Publishing*, Limited. https://books.google.com.ec/books?id=U_X9zgEACAAJ

Calle-González, J. L. (2020). *Metodología para creación de contenido accesible en WordPress para productores de contenido*.

Espinosa, S. (2019). *Evaluación del uso del protocolo QUIC en internet*. https://e-archivo.uc3m.es/handle/10016/29744

Fernández, F. M., Zverev, M., Garrido Ortiz, P., Juárez Rodríguez, J. R., Bilbao Ugalde, J., & Agüero Calvo, R. (2021). *Evolución del Stack IoT: MQTT sobre QUIC*.

Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., & Berners-Lee, T. (1999). RFC2616: Hypertext Transfer Protocol – HTTP/1.1. *RFC Editor*.

Henríquez, R. A. (2017). *Descripción del protocolo HTTP y análisis de sus transferencias mediante metalenguajes*.

Iglesias, C., & Guaman, N. (2021). *Análisis de velocidad de acceso a sitios web comparando protocolo TCP tradicional con SSL vs protocolo QUIC*.

Ke, P. F., Lau, Y. M., & Hanley, D. V. (2023). Is Web3 Better Than Web2 for Investors? Evidence from

Domain Name Auctions.

Kyaw, N. N. (2019). Analysis and simulation of hypertext transfer protocol at the application layer of the internet. *International Journal of Scientific and Research Publications (IJSRP)*, 9(1), 10–29322.

Morocho Troya, E. O. (2022). *Desarrollo de los módulos e-learning y evaluaciones como componentes del entorno virtual de aprendizaje integrado (EVAI) para la empresa IEREC*.

Murthy, A. A., John, P. M., & Kasturi Nagappasetty, R. M. B. (2023). Hypertext transfer protocol performance analysis in traditional and software defined networks during Slowloris attack. *International Journal of Electrical & Computer Engineering (2088-8708)*, 13(4).

NetApplications.com. (2017). *Market Share Statistics for Internet Technologies*. https://n9.cl/1fjc5

Oliveira, A. T. de. (2020). *Uma análise experimental de desempenho do protocolo QUIC*. https://repositorio.ufpe.br/handle/123456789/37646

Priego, L. (2018). *Estudio del protocolo TLS (Transport Layer Security)*. http://hdl.handle.net/10609/81045

Reese, W. (2008). Nginx: the high-performance web server and reverse proxy. *Linux Journal*, 2008(173), 2.

Rescorla, E. (2018). The Transport Layer Security (TLS) Protocol Version 1.3 (Issue 8446). *RFC Editor*. https://doi.org/10.17487/RFC8446

Rodríguez, R. (2020). *Desarrollo de nuevos modelos de interacción usuario-ecommerce: integración de ecommerce en chatbot vía API REST*.

Romero, J. C. (2020). *TLS 1.3*. http://hdl.handle.net/10609/126747

Rueda, E. E. (2019). *Cifrado con el protocolo ssl/tls y el rendimiento de sitios web. caso: empresa web-out, 2018–2019*.

Sandra, H. (2022). Performance Analysis of Openlitespeed and Apache Web Servers in Serving Client Requests. Knowbase: *International Journal of Knowledge in Database*, 2(2), 114–129. https://ejournal.iainbukittinggi.ac.id/index.php/ijokid/article/view/5306

Thomas, L., Dubois, E., Kuhn, N., & Lochin, E. (2019). Google QUIC performance over a public SATCOM access. *International Journal of Satellite Communications and Networking*, 37(6), 601–611.

Tomás, E. (2021). *Estudio comparativo de las versiones más recientes de HTTP*. https://riunet.upv.es:443/handle/10251/172835

Vega, D. (2014). *Estudio de prestaciones del protocolo HTTP versión 2*. https://e-archivo.uc3m.es/handle/10016/27301