Security Review and Performance Analysis of QUIC and TCP Protocols

Conference Paper · October 2022

DOI: 10.1109/ISCTURKEY56345.2022.9931821

Aynur Koçak
Gazi University
8 PUBLICATIONS **16** CITATIONS
SEE PROFILE

Mustafa Alkan
Gazi University
44 PUBLICATIONS **228** CITATIONS
SEE PROFILE

Security Review and Performance Analysis of QUIC and TCP Protocols

Sercan ORAN
Information Security Engineering
Gazi University
Ankara/Türkiye
sercan.oran@gazi.edu.tr
0000-0002-3652-715X

Aynur Koçak
Electrical Electronics Engineering
Gazi University
Ankara/Türkiye
aynurkocak@gazi.edu.tr
0000-0001-9647-7281

Mustafa ALKAN
Electrical Electronics Engineering
Gazi University
Ankara/Türkiye
alkan@gazi.edu.tr
0000-0002-9542-8039

2022 15th International Conference on Information Security and Cryptography (ISCTURKEY) | 978-1-6654-5603-6/22/$31.00 ©2022 IEEE | DOI: 10.1109/ISCTURKEY56345.2022.9931821

***Abstract*— In this study, the development process and basic features of the QUIC protocol have were introduced and the tests and analysis of the QUIC protocol in network environments and network devices were carried out comprehensively. On the client side, a test scenario was created using Curl and Quiche libraries, on the server side with Cloudflare. All network traffics were analyzed by recording pcapng packet files with Wireshark application. QUIC and TCP protocol page load time, throughput, performance and security analyzes have been conducted. In addition, Flow Control, Connection Carrying and Load, Reliable and Multiple Connections, Head-of Line Blocking and Congestion Control applications were analyzed comparatively within the scope of TCP and QUIC protocols.**

As a result of the analysis, it has been observed that the QUIC protocol outperformed the TCP protocol in the page load times, throughput and security applications. However, it has been found out that the QUIC protocol consumes more processing power than the TCP protocol with these applications. Although QUIC is a new protocol, it is considered that its usage in network traffic will continue to increase with both in performance and security.

***Keywords*— TCP, QUIC, HTTP stack, Transport layer performance, Network security.**

I. Introduction

The most important task of a transport protocol is to provide a stable connection between two network devices. These network devices include routers, switches, firewalls, etc., such as hosts or devices. The transport protocol is responsible for logical communication between applications running on different computers. There are two important protocols, TCP (Transmission Control Protocol) [1], which is used for connection services, and UDP (User Datagram Protocol) [2] for connectionless services, respectively. As a connection-oriented protocol, TCP provides a reliable end-to-end connection. It also uses congestion control mechanism to prevent buffer overflow on the receiving side. TCP protocol is the most used and distributed protocol on the internet. TCP, which has existed since the use of the Internet, has been in use for over 40 years. However, the TCP protocol was not actually developed with maximum throughput and security in mind. Over time, TCP protocol has been developed and new versions have been released to solve some problems and even introduce new performance and security features [3], [4].

The QUIC protocol was first revealed by Google engineers in 2012 [5]. Then, in 2016, a working group was established by IETF engineers to standardize the QUIC protocol [6]. This working group has worked to make HTTP/2 compatible with QUIC. This new batch version is named HTTP/3 and standardized and published as RFC-9000 in May 2021 [7]. Also, draft QUIC version 2 was published on January 22, 2022 [8]. In this context, with the emergence of the QUIC protocol, security and performance analyzes need to be made in many different network environments. In this study, test analyzes of the QUIC protocol conducted in different test environments were examined.

In the study by Das, QUIC performance over 500+ web pages limited bandwidth and high RTT was analyzed using QuicShell and compared to TCP. They noticed that QUIC performed better on smaller web pages with fewer files and resources [9].

In the study by Jager et al, they analyzed the security of QUIC and TLS1.3 from the perspective of the attackers. In TLS1.2, the Bleichenbacher attack and its variants have shown that it quickly guesses the server's key and destroys its security [10]. They stated that, thanks to TLS1.3 in the QUIC protocol, the support of the PKCS1v1.5 standard, which caused security vulnerabilities in TLS1.2, was removed, theoretically, it no longer poses a threat with such attacks [11].

In the work by Kakhki et al. performed extensive experiments with QUIC in a variety of network conditions, including desktop and mobile, wired and wireless environments. QUIC has generally performed better than TCP. However, performance issues related to frame sizes, reordered packets, and multiplexing of large numbers of small files, and decreased performance of QUIC on mobile devices have been found [12].

In this context, the general purpose of our work is to contribute to the people who will use this protocol in the future by introducing and analyzing the newly emerged QUIC protocol and therefore the HTTP/3 stack. Another purpose of this study is to help users who will study the QUIC protocol independently from large companies. It is recommended to perform these performances and security analysis on large active networks.

II. TRANSPORT LAYER PROTOCOLS

The transport layer is the layer that establishes a connection between the network layer and the application layer. The transport layer is responsible for transmitting data in segments by dividing it into small packets. It is also responsible for controlling the data coming from the upper layer and combining the segments sent to the opposite party correctly.

A. Transmission Control Protocol

TCP one of the most important protocols of the TCP/IP model, is designed as a reliable and full-featured transport protocol that ensures that all data reaches the destination [13]. For this protocol to work, devices must do a triple handshake among themselves. When data is sent with the TCP protocol, the target device sends a confirmation message that it has received the data without any problems. If the confirmation message is not received within a certain time, the package is sent again.

TCP protocol, to provide transport security and to send the obtained data, divide the data into packets of the most appropriate size, receive an acknowledgment message for each packet sent and send the packet again if the confirmation message is not received, send an acknowledgment message for the received packet, add the sequence number to the packet header information and recover the corrupted packets. It performs steps such as distinguishing, reordering the packages if they reach the server in different orders, and eliminating the extra packages if the same package arrives in more than one number [14].

In order to keep track of the status and communication information of a data exchange, a communication setup between the sender and receiver in the TCP protocol must be established. For this reason, TCP protocol communication is also called connection-oriented protocol. The TCP protocol is a connection-oriented protocol that establishes a persistent session between source and destination devices before sending any data packets. During any session setup, the network devices decide the amount of traffic to send at a given time, allowing the connection data between two network devices to be managed. Due to any problem, it is possible for the packets to be completely corrupted or lost while transferring over network devices. The TCP protocol ensures that every packet sent by the source device reaches the destination device. Data may arrive at the servers in the wrong order, as the networks that provide communication can use multiple paths with different transfer rates. The TCP protocol can enumerate and sequence packets so that these packets are reassembled in the correct order. Network devices have limited resources such as memory and processing power. When the TCP protocol realizes that these resources are overloaded, it can ask the transmitting device to reduce the data flow rate. The TCP protocol does this by regulating the amount of data the source device sends. TCP protocol flow control can eliminate the need to resend data when the receiving devices' resources are overloaded.

The TCP protocol is a stateful protocol, meaning it monitors the status of the communication session. To monitor the status of a session, TCP records what information it has sent and what information has been acknowledged. A stateful session begins with session establishment and ends with session termination. HTTP, HTTPS, FTP, SMTP and SSH etc. protocols can use TCP services by sending the data stream to the transport layer. A TCP packet adds 20 bytes of overhead when encapsulating application layer data. The "TABLE I" below shows the packet header in a TCP header [15].

TABLE I. TCP HEADER FORMAT

Source Port(16)			Target Port(16)
Row Number(32)			
Confirmation Number(32)			
Header Length(4)	Split (4)	Control Bits(6)	Window(16)
Checksum(16)			Urgent(16)
Options (0 or 32 if any)			
Options (Application Layer Data (Size varies)			

B. User Datagram Protocol

UDP, which is a simple and fast protocol, is used to transmit data from one point to another [16]. UDP packet headers are faster because they contain less information than other communication protocols. However, it is not reliable because it does not include control operations. This protocol is used in operations such as SNMP, VoIP, DHCP, TFTP and DNS service, which are network management protocols. The UDP packet header is 8 bytes long. The packet header consists of the port of the device sending the data, the port of the destination device to which the data will reach, the length of the message and the checksum. UDP is a lightweight transport protocol that segments and reassembles data in the same way as TCP, but without the reliability and flow control of TCP. The UDP protocol has basic features such as regenerating data in the order they are received, not resending lost segments, no session management, and not informing the sending device about resource availability. UDP is a stateless protocol, meaning neither the client nor the server monitors the state of the communication session. When using UDP as the transport layer, reliability must be provided by the application when needed.

One of the most important requirements of transmitting live video and audio over the network is to keep data flowing fast. Live video and audio applications can tolerate some loss of data minimal or undetected, which would be better suited for the UDP protocol. Communication blocks in UDP are called datagrams or segments. These datagrams are sent by the transport layer protocol on a best-effort basis. UDP packet header information is much simpler than TCP packet header information. Because it has four fields of only 8 bytes. The "TABLE II" below shows the fields in a UDP packet header.

TABLE II. UDP HEADER FORMAT

Source Port(16)	Target Port (16)
Lenght(16)	Checksum(16)
Application Layer Data (Size varies)	

C. Quick UDP Internet Connections

The QUIC protocol stands for Fast UDP Internet Connections and was developed by Google. QUIC provides security and reliability with reduced connection and transfer latency. Google and many service providers have installed the QUIC protocol widely on their servers and it is used at certain rates. In Cloudflare Radar's report dated February 22, 2022, HTTP/1.X, HTTP/2 and HTTP/3 stacks that provide communication in internet traffic are used at 8\%, 68\% and 24\% respectively, TLS1.2, TLS1.3 and QUIC protocols were reported to be used in 13\%, 63\%, and 24\%, respectively [17].

Unlike other transport layer protocols, QUIC is also included in the application layer. The QUIC protocol was developed over UDP for use on existing network devices. QUIC traffic on network devices is carried out as UDP traffic. One of the main ideas in the introduction of the QUIC protocol was to overcome the TCP limitations highlighted earlier. The UDP protocol, which is the main transport protocol that also builds on the QUIC protocol, is unreliable. The UDP protocol does not provide the features required for a reliable connection. In this context, the QUIC protocol offers new features to solve TCP problems. HTTP2 stack connection security features realized with TCP protocol over TLS, HTTP3 stack with QUIC is implemented over UDP protocol, so connection security is provided at the application layer. QUIC, a protocol that can do many features on its own, offers various innovation opportunities in packet headers that are not possible to achieve with existing protocols.

A newly developed congestion control algorithm is used for the congestion control of the QUIC protocol [18]. According to the congestion control algorithm of the TCP protocol, the control algorithm of the QUIC protocol has been further developed. For example, each original or retransmitted packet has a unique sequence number. This simple sequence number solves the ambiguity problem in the TCP protocol by distinguishing the original and retransmitted packets in the QUIC protocol [19]. Thanks to the forward error correction mechanism in the QUIC protocol, these packets can be recovered without having to worry about retransmitting lost packets [20].

Compared with the QUIC protocol and the TCP protocol over TLS, that is, the HTTP2 stack, reduced connection establishment time (0-RTT), multiplexing, connection migration, secure communication (security transport) provide advantages.

The QUIC protocol greatly reduces the latency during connection establishment by reducing RTT times in the triple handshake stage compared to the HTTP2 stack.

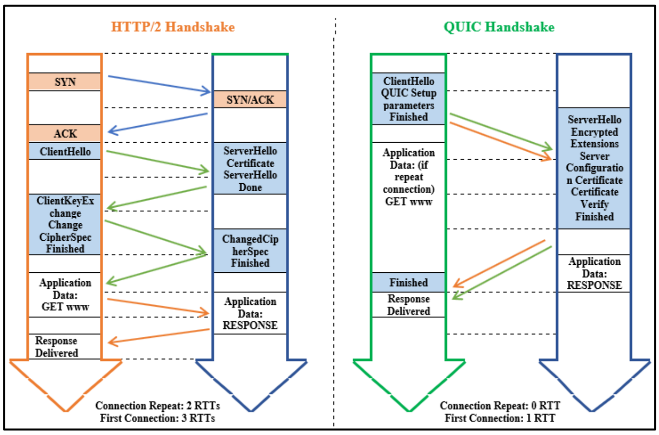

HTTP/2 Handshake
SYN
SYN/ACK
ACK
ClientHello
ServerHello Certificate ServerHello Done
ClientKeyExchange Change CipherSpec Finished
ChangedCipherSpec Finished
Application Data: GET www
Application Data: RESPONSE
Response Delivered
Connection Repeat: 2 RTTs
First Connection: 3 RTTs
QUIC Handshake
ClientHello QUIC Setup parameters Finished
ServerHello Encrypted Extensions Server Configuration Certificate Certificate Verify Finished
Application Data: (if repeat connection) GET www
Finished
Application Data: RESPONSE
Response Delivered
Connection Repeat: 0 RTT
First Connection: 1 RTT

Fig. 1. HTTP/2 and QUIC handshake comparison

HTTP2 stack provides faster communication by sending multiple packets with a single TCP connection. If a single file or resource drops during the communication, this lost file or resource needs to be retransmitted, and the TCP connection is completely stopped until this lost file/resource comes back. This is called the HOLB problem. As this packet loss rate increases, HTTP2 performs worse in communication. The HOLB problem of TCP is shown in "Fig.2".

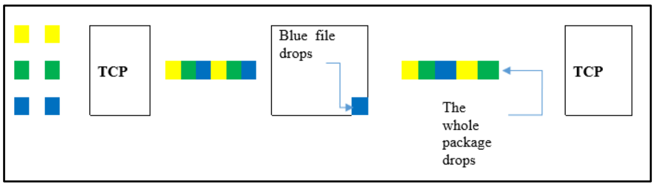


Fig. 2. Head-of line blocking problem (HOLB)

Unlike the HTTP2 stack, the QUIC protocol establishes multiple independent flows between the server and the client. If the connection breaks in one of the streams, it drops from one of the files or resources that make up the packet, that stream only has to wait for the missing link to be retransmitted and the other files or resources continue to communicate. This situation is illustrated in "Fig.3."

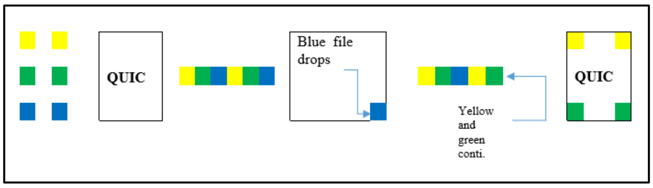


Fig. 3. QUIC Head-of line blocking solution

The QUIC protocol supports secure data transmission by incorporating TLS encryption. TLS is used as an additional header in the TCP protocol. Packets use separate protocols to perform their operations. By combining this handshake as a part of its own handshake, QUIC provides authentication of endpoints and negotiates all encryption parameters within itself. In this communication, it always ensures that the identity is authenticated and encrypted. At the same time, one more RTT is reduced. In addition, QUIC firewalls also encrypt their metadata so that intermediate devices such as proxies do not change the connection.

A 64-bit connection identifier (CID - Connection ID) is defined in QUIC connections. Changing any of the parameters of the destination IP address, source port number and destination port number in the TCP protocol causes the connection to be dropped and the session no longer active. "Fig.4." shows the connection transition problem mentioned for TCP.

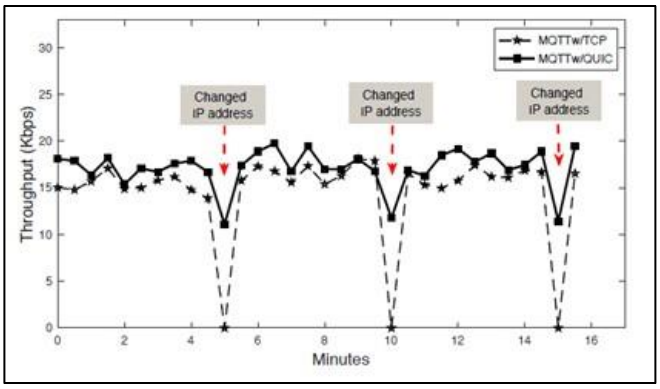
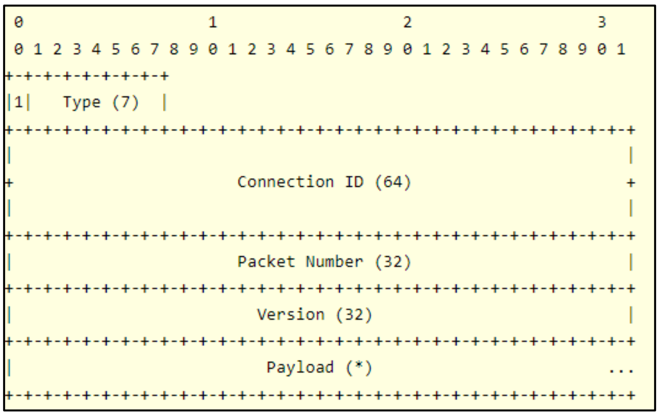


Fig. 4. TCP connection relays problem [21]

QUIC keeps the connection ID the same thanks to the CID, and does not break the connection even if there is a change in any of these parameters. If the client's IP address changes, the connection from the old connection ID and the new IP address continues without any interruption. In this way, QUIC offers a solution to the connection problem experienced in 4G/WIFI transitions, especially on mobile platforms. Thanks to this important feature enabled by QUIC, seamless and transparent connection transition between different networks is provided without the need to create a completely new connection.

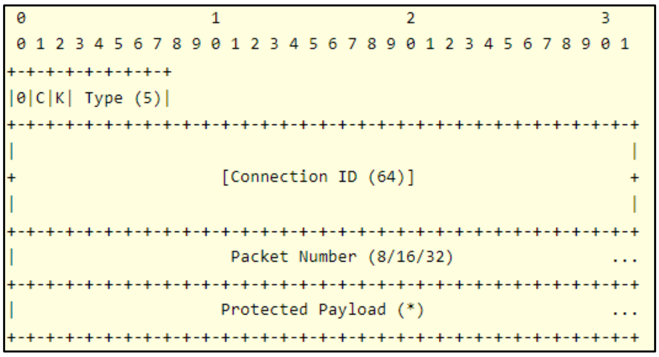
In the QUIC protocol, there are usually long and short headers, version agreement packets, initial and 1-RTT and 0-RTT packets, etc. packet formats are used. QUIC has two packet header formats, long and short header format. As the name suggests, the long header format is large in size and the short header format is small in size. In long title format; version, target link id, source link id, header form, hard bit etc. contains information [22].



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+
|1|   Type (7)  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                       Connection ID (64)                      +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Packet Number (32)                      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                         Version (32)                          |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Payload (*)                        ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fig. 5. QUIC long title format [23]

"Fig.5." shows the long title format in the QUIC protocol. Long headers are packets transmitted before initial packets are created with 1-RTT switches. First, the keys are agreed upon. Then once the communication is complete the sender starts using the short header again. Version deal packages etc in QUIC long header format. It allows special packages to be represented in a uniform fixed-length package format. The QUIC short packet format is the most widely used and is used after 1-RTT key agreement. Short header packet; consists of the destination link ID, packet number, and protected payload.



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+
|0|C|K| Type (5)|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
+                     [Connection ID (64)]                      +
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      Packet Number (8/16/32)                ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Protected Payload (*)                   ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Fig. 6. QUIC short title format [23]

III. MATERIAL AND METHOD

Testing was done using an HP desktop computer running Linux 4.15.0-129-generic64-bit Ubuntu 16.04 LTS. The hardware of the computer includes Intel(R) Core (TM) i7-3520M CPU @ 2.90GHz (with SSE4.2) and 8 GB RAM. Dumpcap (Wireshark) 3.2.7 (Git v3.2.7 packaged as 3.2.7-1~ubuntu16.04.0+wiresharkdevstable1) application is used. The files used in the QUIC and TCP protocols were the same size. Wireshark application was used to analyze the files in question [24]. As a result, curl and quiche has been used on the client side and Cloudflare on the server side.

IV. RESULTS AND DISCUSSION

We examined and evaluated the concepts of page load time, throughput and security analysis between QUIC and TCP protocols.

A. Page Load Lime

In our page load test, we used URL data in size of 27kB, 48kB, 616kB, 653kB,904kB, 663kB, 103kB, 642kB, 120kB, 639kB, and 623kB. As shown in "Fig. 7." QUIC Protocol performed better than TCP when all data were tested. The QUIC protocol performed approximately 8\%-23\% better than the TCP protocol in page load times.

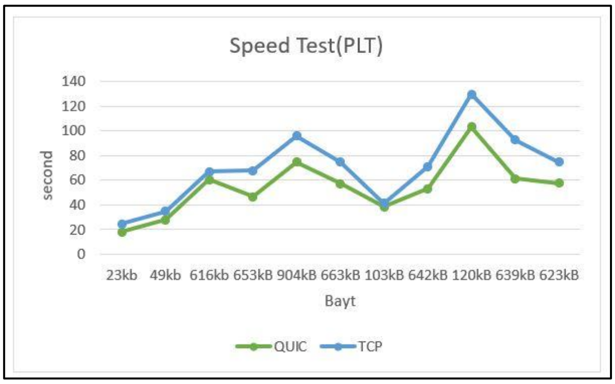


Fig. 7. Speed test result of QUIC and TCP protocols

B. Throughput

In the throughput test, URL data of 27kB, 48kB, 616kB, 653kB, 904kB, 663kB, 103kB, 642kB, 120kB, 639kB and 623kB were used. As shown in "Fig. 2" QUIC Protocol performed better than TCP when all data were tested. the QUIC protocol outperformed the TCP protocol by approximately 12\%-27\%. Examining the reason why the QUIC protocol outperformed the TCP protocol, QUIC achieved better throughput than the TCP protocol by reducing the acknowledgment frequency and using as large a packet size as possible. Better performance is directly proportional to the package size. However, when the packets are large, it can cause some network traffic drops. As the loss rate increases, the connection becomes congested and gradually reduces the data rate in this TCP protocol. However, due to the fact that it is built on UDP and the update mechanism, QUIC's performance has increased by 12\% over TCP.

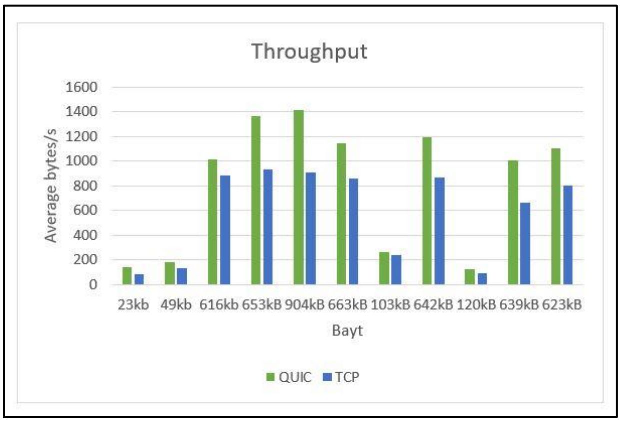


Fig. 8. Throughput test results graph of QUIC and TCP protocols

C. Securıty Revıew

Today, we store almost all of our information on network devices. Confidentiality, integrity and accessibility, which are the basic elements of information security, must be ensured in order to store this information without any problems. In this section, an analysis has been made on how QUIC and TCP protocols provide the mentioned information security elements in network traffic. As mentioned before, TLS protocol works independently of TCP protocol. The QUIC protocol designers have taken this stand-alone operation to the next level. In the HTTP/3 stack, encryption is performed in QUIC. While early versions of Google's QUIC were implemented separately, the standardized QUIC implemented existing TLS 1.3 by incorporating it. This case is shown in "Fig. 9.".

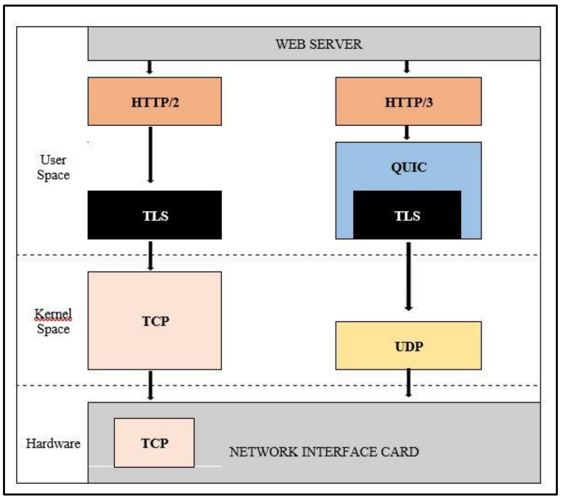


Fig. 9. *HTTP/3 stack*

The QUIC protocol uses TLS in itself. While TLS1.3 can run independently over TCP, QUIC encapsulates TLS 1.3 instead [25]. In other words, when QUIC is used, packages will definitely pass TLS. Developed on QUIC, HTTP/3 completely encrypts packets. In addition, transport layer information such as packet numbers, sequence numbers, ACK numbers can no longer be read by attacking people and institutions by listening to data traffic in QUIC. QUIC uses unified encryption to reduce connection latency. QUIC provides the cryptographic handshake with the CRYPTO framework [26].

As seen in “Fig. 9.” QUIC uses the TLS1.3 handshake as in TCP [27]. However, the QUIC protocol encrypts the packets itself because it absorbs the TLS protocol itself. However, it is not the case with the TCP protocol. Packets first enter the TLS protocol and then pass through the TCP protocol. This difference is an important change for the QUIC protocol, which will be used in the future.

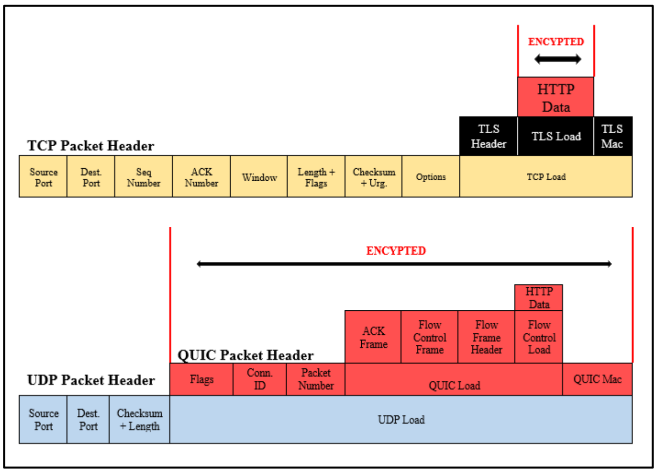


Fig. 10. TCP+TLS and QUIC packages

“Fig. 10.” Packages of TCP and QUIC protocols are examined in detail. When TCP packets are analyzed, it is seen that packet numbers, ACK numbers, windows and options are not encrypted. When QUIC packets are analyzed, it is observed that almost all traffic is encrypted. This important difference provides a number of advantages and disadvantages stated in the following items in the QUIC protocol .

The first of the advantages of QUIC is that the QUIC protocol has become more secure for its users. Reading clear texts flowing on the Internet is blocked by QUIC and almost all network traffic is encrypted. For this reason, it has become difficult for attackers and people watching the network to listen. Recent research on plaintext used in network traffic has shown how dangerous the plaintext option of HTTP/2 can be [28]. Second, QUIC is faster. In TCP protocol, it performs handshake separately together with TLS. The QUIC protocol absorbs TLS in itself, reducing one round trip (RTT). Third, the QUIC protocol will be easy to develop. The inner workings of TCP are observed but cannot be observed as the QUIC is fully encrypted so it will be easier to update new versions of QUIC.

However, besides these benefits, comprehensive encryption also has some potential disadvantages. First, many institutions may not allow the use of QUIC. Organizations may not want QUIC protocol support in their firewalls, as it will not be easy to detect malicious traffic. Since the packets are encrypted, the latency and packet loss measurements will be difficult and malicious traffic will not be detected. Therefore, internet service providers and intermediate networks will be able to prevent this. Second, the encryption overhead of the QUIC protocol is high. The QUIC protocol encrypts packets with TLS. Therefore, the use of the QUIC protocol may not be appropriate in scenarios where CPU usage is low. Third, there is fear that QUIC will make the web more centralized and monitor and analyze all traffic. So QUIC can cause problems as Google will not share any of it with others while providing full access to the data. All this can make QUIC difficult to use universally.

The QUIC protocol differs from TCP in some new features. In “TABLE III” these features are examined comparatively.

TABLE III. COMPARATIVE ANALYSIS OF TCP AND QUIC PROTOCOLS

Problems	TCP	UDP	QUIC
Multiple Connection to Server	✓	**N/A**	✓
Head of Line Blocking	**X**	**X**	✓
Connection Load	**X**	**X**	✓
Connection Transport	**X**	**X**	✓
Congestion Control	✓	**X**	✓
Flow Control	✓	**X**	✓
Reliable Connection	✓	**X**	✓

V. CONCLUSION

Considering that the use of emerging and disruptive technologies such as Artificial Intelligence, IoT, Machine Learning etc. will become increasingly widespread, it is an inevitable fact that communication between devices and systems in the future should be realized in a fast and secure way. In this study, performance analysis and security analysis were made between the QUIC protocol, which was recently standardized and developed by the IETF, and TCP, which is currently used as the most common transport protocol, and suggestions were presented in this direction. The QUIC protocol has solved the line blocking, connection handling, connection load issues and brought new improvements to the server in multi-connection, reliable connection, flow control and congestion control, which is also used in the TCP protocol. Due to these important changes, the use of QUIC protocol in network traffic will increase gradually, and it is considered that it will be appropriate for public institutions and companies in the private sector to provide QUIC support on their own network devices in the future.

REFERENCES

[1] S. M. Bellovin, “Security problems in the TCP/IP protocol suite,” *ACM SIGCOMM Comput. Commun. Rev.*, vol. 19, no. 2, pp. 32–48, Apr. 1989, doi: 10.1145/378444.378449.

[2] L.-Å. Larzon, M. Degermark, and S. Pink, “UDP Lite for Real Time Multimedia Applications,” 1999.

[3] F. Gont and A. Yourtchenko, “On the Implementation of the TCP Urgent Mechanism,” Jan. 2011, doi: 10.17487/RFC6093.

[4] F. Gont and S. Bellovin, “Defending against Sequence Number Attacks,” Feb. 2012, doi: 10.17487/RFC6528.

[5] G. Florian, “QUIC - Quick UDP Internet Connections Florian,” *Futur. Internet Innov. Internet Technol. Mob. Commun.*, no. September, pp. 1–7, 2016, doi: 10.2313/NET-2016-09-1.

[6] Y. Cui, T. Li, C. Liu, X. Wang, and M. Kuhlewind, “Innovating transport with QUIC: Design approaches and research challenges,” *IEEE Internet Comput.*, vol. 21, no. 2, pp. 72–76, 2017, doi: 10.1109/MIC.2017.44.

[7] Saverio Massano, “POLITECNICO DI TORINO A new One-Way Delay measurement system for QUIC protocol Candidate: Saverio Massano.”

[8] M. Kosek, T. Shreedhar, and V. Bajpai, “Beyond QUIC v1: A First Look at Recent Transport Layer IETF Standardization Efforts,” *IEEE Commun. Mag.*, vol. 59, no. 4, pp. 24–29, Apr. 2021, doi: 10.1109/MCOM.001.2000877.

[9] I. S. C. Bachelor, “Evaluation of QUIC on Web Page Performance,” *MIT Libr.*, pp. 1–172, 2014.

[10] C. Meyer, J. Somorovsky, E. Weiss, J. Schwenk, H. Görtz, and S. Schinzel, “Revisiting SSL/TLS Implementations: New Bleichenbacher Side Channels and Attacks,” Accessed: Feb. 05, 2022. [Online]. Available: https://www.usenix.org/conference/usenixsecurity14/technical-sessions/presentation/meyer.

[11] T. Jager, J. Schwenk, and J. Somorovsky, “On the security of TLS 1.3 and QUIC against weaknesses in PKCS#1 v1.5 encryption,” *Proc. ACM Conf. Comput. Commun. Secur.*, vol. 2015-Octob, pp. 1185–1196, 2015, doi: 10.1145/2810103.2813657.

[12] A. M. Kakhki, S. Jero, D. Choffnes, C. Nita-Rotaru, and A. Mislove, “Taking a long look at quic: An approach for rigorous evaluation of rapidly evolving transport protocols,” *Proc. ACM SIGCOMM Internet Meas. Conf. IMC*, vol. Part F1319, pp. 290–303, 2017, doi: 10.1145/3131365.3131368.

[13] S. Rajab, “Performance testing TCP and QUIC,” no. May, 2016.

[14] R. Çölkesen, B. Üniversitesi, and B. Örencik, “Bilgisayar Haberleşmesi ve Ağ Teknolojileri,” Accessed: Feb. 19, 2022. [Online]. Available: http://www.papatya.gen.tr.

[15] K. Ahsan and D. Kundur, “Practical Data Hiding in TCP/IP,” *Gov. Ontario*, 2000.

[16] P. P. K. Lam and S. C. Liew, “UDP-Liter: An Improved UDP protocol for real-time multimedia applications over wireless links,” *1st Int. Symp. Wirel. Commun. Syst. 2004, Proc. ISWCS ’04*, pp. 314–318, 2004, doi: 10.1109/ISWCS.2004.1407260.

[17] “Cloudflare Radar.” https://radar.cloudflare.com/ (accessed Feb. 20, 2022).

[18] Batenburg, “Performance of DNS over QUIC - University of Twente Student Theses.” https://essay.utwente.nl/89441/ (accessed Jun. 27, 2022).

[19] A. Gurtov, “Resolving Acknowledgment Ambiguity in non-SACK TCP,” no. February 2004, 2014.

[20] F. Michel, Q. De Coninck, and O. Bonaventure, “Quic-fec: bringing the benefits of forward erasure correction to quic,” *2019 IFIP Netw. Conf. IFIP Netw. 2019*, May 2019, doi: 10.23919/IFIPNETWORKING.2019.8816838.

[21] P. Kumar and B. Dezfouli, “Implementation and Analysis of QUIC for MQTT,” Oct. 2018, [Online]. Available: http://arxiv.org/abs/1810.07730.

[22] J. Reynders, “QUIC Insight (A thesis presented for the degree of Bachelor of Computer Science),” *A thesis Present. degree Bachelor Comput. Sci.*, vol. A thesis p, 2018.

[23] J. Iyengar and M. Thomson, “RFC 9000: QUIC: A UDP-Based Multiplexed and Secure Transport,” 2021, Accessed: Jun. 27, 2022. [Online]. Available: https://www.rfc-editor.org/info/rfc9000.

[24] U. Lamping, R. Sharpe, and E. Warnicke, “Wireshark User’s Guide - v1.11.3-rc1-1721-gdd4e5fc for Wireshark 1.11,” 2004.

[25] “HTTP/3: Performance Improvements (Part 2) — Smashing Magazine.” https://www.smashingmagazine.com/2021/08/http3-performance-improvements-part2/ (accessed Feb. 12, 2022).

[26] J. Iyengar and M. Thomson, “Do Not Deploy This Version of Quic,” no. July, pp. 1–207, 2021.

[27] M. Thomson and S. Turner, “RFC 9001 Using TLS to Secure QUIC,” pp. 1–52, 2021.

[28] Z. Qian, Z. M. Mao, and Y. Xie, “Collaborative TCP sequence number inference attack - How to crack sequence number under a second,” *Proc. ACM Conf. Comput. Commun. Secur.*, pp. 593–604, 2012, doi: 10.1145/2382196.2382258.