Journal of ELECTRICAL ENGINEERING, VOL 72(2021), NO1, 20–28

sciendo

PAPERS

QUIC transmission protocol: Test-bed design, implementation and experimental evaluation

Ala' Khalifeh[1*], Ma'moun Mansour[2], Sahel Alouneh[3,4]

With the ever increasing demand for higher speed internet connectivity that can fulfil the application continuous need for higher bandwidth Google being the pioneer in many web-based services has launched a new UDP-based protocol named quick UDP internet connections (QUIC), which aims at providing faster data delivery without requiring upgrades or modifications to the network infrastructure. The goal of this paper is to provide an overview about QUIC protocol, propose the design and implementation of a test-bed, that is used experimentally to evaluate QUIC protocol under different network conditions and scenarios. In particular, the performance advantage of QUIC in terms of delay and throughput are examined taking into account different network conditions that resemble the real internet environment. Two scenarios are proposed, the first one investigates the protocol performance under a controlled network environment, while the second one tests the protocol in a real uncontrolled network. To achieve that, a test-bed is proposed and implemented that emulates the network impairments encountered in real-network such as packet loss, bit errors, and bandwidth limitation in a controlled manner. After that, QUIC is tested in real operational wired and wireless networks. In both scenarios, QUIC outperforms TCP in terms of delay, which strengthens QUIC position for being a potential alternative to TCP.

K e y w o r d s: Google, TCPIP, web services, broadband communication, performance evaluation, QUIC, wireless networking

1 Introduction

Internet is the most evolving and fastest changing technology in history. The demand of Internet is increasing day by day, and the ambitions and requirements are increasing exponentially. Billions of devices and computers are connected to the Internet nowadays, beside these devices, there is a change in the Internet content paradigm driven by video and audio services, where new technology concepts appear like cloud computing, social media, Internet of Things (IoT) and big data. The presence of these platforms and technology imposes an obligation on the research and development community to improve the Internet infrastructure, and leverage the current Internet protocols to cope with the increased demand on high-speed data delivery and access. Several attempts on designing application and data link level solutions to mitigate packet loss and bit errors have been made. These attempts consider the wired and wireless Internet connections for the purpose of improving the transmission speed and quality, but without changing the current transmission control protocols[1-5]. However, to obtain significant results, more research has to be done in the data transmission process.

This paper proposes a thorough examination and comprehensive comparison between QUIC and TCP through testing the two protocols in a test-bed, which is designed to achieve the goals of this study, and are demonstrated in two scenarios: the first one examines the protocol under a controlled network that is an unrestricted network of any outside effects, and has a network emulator that controls the network impairments such as packet loss, bandwidth limitation and bit errors rate. The second scenario tests the protocol under uncontrolled or real production networks, which is the case of the network that may contain any types of real network error factors. Therefore, it should show the capabilities of QUIC over TCP in real production networks, and highlight the potential of QUIC to replace TCP. The contributions of this paper are summarized as follows:

- A new testbed is proposed with a simple and clear description of its components, and implementation steps. Further, the paper highlights some of the implementation practical challenges faced, which will benefit the research community and assist in testing and developing QUIC protocol.
- QUIC protocol performance is evaluated under different controlled network impairments thus emulating several real network conditions and scenarios.
- QUIC protocol efficiency in real-production network is assessed, and its seamless integration with the current network infrastructure is examined.

[1] School of Electrical Eng. and Information Technology, German Jordanian University, Amman, Jordan; ala.khalifeh@gju.edu.jo, [2] School of Electrical Engineering and Information Technology, German Jordanian University, Amman, Jordan, mamounm78@gmail.com, [3] College of Engineering, Al Ain University, Abu Dhabi, UAE, sahel.alouneh@aau.ac.ae, [4] School of Electrical Engineering and Information Technology, German Jordanian University, Amman, Jordan, Sahel.alouneh@gju.edu.jo, * Correspondence: ala.khalifeh@gju.edu.jo

DOI:10.2478/jee-2021-0003, Print (till 2015) ISSN 1335-3632, On-line ISSN 1339-309X

2 Related work

The aim of this section is to summarize the most related works in the literature. In [6], QUIC is compared with SPDY and HTTP /1. 1 as these protocols are multiplexing protocols, and an evaluation of the strength and weakness of QUIC was introduced by loading the Alexa U. S Top 500 web sites in real web browser. Then, measuring the load time over several network configurations of variable bandwidth and round-trip time. This has been done by implementing tools for emulating web sites traffic over QUIC, and a tool for congestion control protocol.

In [7], the work checked experimentally if QUIC could be safely deployed, with an evaluation of the web page load time in comparison with SPDY and HTTP. The work concentrated on the modification of network bandwidth and packet loss with Forward Error Correction (FEC) switched on or off. In this study, there was no test for wireless networks and no consideration for real production networks. Further, Google has recently removed FEC from QUIC, because it did not show sufficient performance benefits, and it is still being under development by Google.

The authors in [8] tested QUIC under Wireless Mesh Network (WMN). Their experimental evaluation depicted that although QUIC outperformed TCP in wired environment, it was not the case in WMN. In the later scenario, QUIC had some sub-optimal interaction with the MAC layer features, such as the frames aggregations, which causes a degradation in its performance. The author proposed an improved version, which outperformed QUIC under WMN. However, in their study, the focused on WMN, which is a specific scenario on wireless networks, while we covered a variety of wireless communication links and technologies that are practical and widely used.

In [9], the authors proposed an application-level data transfer protocol called SABUL for reliability, high performance, fairness and stability, which is achieved by data transfer over UDP, and returned a control message over TCP. SABUL uses a rate-based technique as a solution for rate based congestion, while QUIC protocol solves congestion by pacing. SABUL also depends only on retransmission techniques for lost packets. The authors in [10] studied the performance of QUIC protocol utilizing the NS3 simulator. Limited scenarios have been tested with no practical test-bed evaluation and implementation.

In [11], the authors discussed QUIC protocol main features and challenges. In particular, the authors described QUIC ability of sending multiple streams of data over a single connection without suffering from the Headof-line Blocking, its congestion control algorithm which is slightly different from TCP, as it adopts the F-RTO and Early retransmission algorithms, and its supports upto 255 ACK ranges, which makes it more robust against packet loss and packets' reordering. Further, the authors highlighted some challenges that faces QUIC such as its ability to distinguish between packet loss that are caused by network congestion and the one that are caused by wireless bit errors, and how the protocol reacts for each loss type. Another challenge is related to the protocol privacy and security, especially that QUIC has many unencrypted information such as the connection identifier, which makes it susceptible to pervasive monitoring attacks. Finally, the authors emphasize the fact that QUIC can be further optimized based on the application specifications and requirements. In the literature, several studies have been conducted in QUIC to analyze its performance or to explore its features and weaknesses. In this paper, we are contributing to this line of research by having an experimental study that is based on a realistic and controlled test-0bed, where the protocol performance can be studied accurately in different network types and scenarios. Furthermore, the paper provides realistic insights for the protocol performance in a real and production networks, which helps the research community in pursing the development work of this promising protocol.

3 QUIC Protocol Description

In this section, a description of QUIC protocol is introduced. QUIC was proposed by Google in 2013 [12], its main goal is to reduce latency by operating on top of UDP. Fig. 1 shows the upper network layers of both SPDY and QUIC. As depicted, the layer of QUIC replaces TCP and Transport Layer Security (TLS), which is used to secure all communications between server and web browsers in SPDY/HTTP2, QUIC layer also includes the security equivalent to TLS in SPDY which is called Crypto[13]. In what follows, the most important characteristics of the QUIC protocol are briefly described.

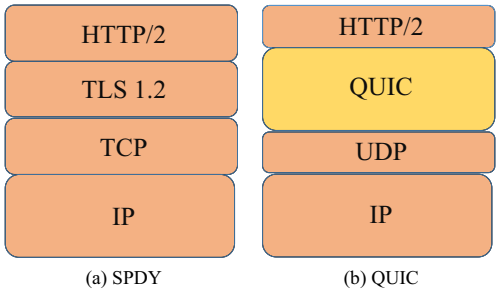

Fig. 1. (a) – SPDY, and (b) – QUIC layers

3.1 Secure Connection Establishment

HTTP over TCP uses TLS to apply security in data transmission, while in QUIC, TLS is replaced by Crypto. The main advantage of Crypto; which keeps pace with the goal of QUIC in latency reduction; is that when client caches information from the server, it can re-establish an encrypted connection with no round trip, thus decreases latency, as shown in Fig. 2. Additionally, Crypto has a greater security level than TLS[12], as it is always encrypted, so the basic unit of transmission is a standard UDP packet.

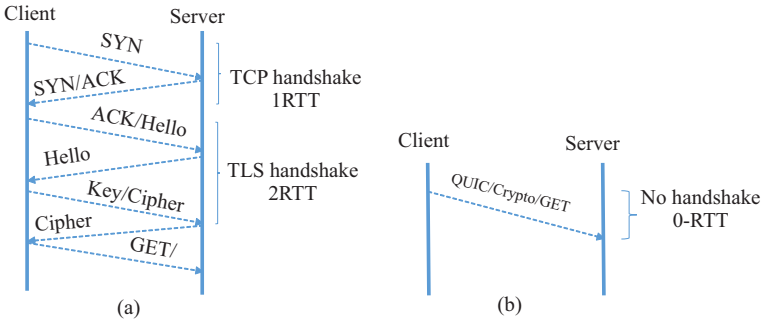

Fig. 2. Connection establishment round trip (a) – 3-RTT TCP and TLS handshake, (b) – 0-RTT in QUIC

3.2 Multiplexing

In SPDY, multiplexing produces Head of Line Blocking (HOLB), because it multiplexes many streams sessions over a single connection, and any packet loss that occurs will block all the streams until a retransmission of the lost packet arrives. As shown in Fig. 3 (a), the stream that consists of packets 2, 3, and 4 and the stream consists of packets 5 and 6 are blocked, because the loss of packet1, the block will take place until the retransmission of packet number 1, on the other hand, QUIC prevention of HOLB occurs because it is built over UDP[14], so it can support out-of-order delivery and has several byte streams, the result is that the only impact of packet loss is for an individual stream, and the stream without loss still can continue to be reassembled and go in progress. Fig. 3 (b) shows that despite the drop of packet number 1, the other streams of packets are received and reassembled by the client, without any delay that may occur in case of waiting the retransmission of the lost packet.

3.3 Connection migration

A major change in connection identification by QUIC is the addition of a 64-bit Connection Identification (CID) at the application layer, which is generated randomly by the client. In TCP, connection is identified by a 4-tuple of source address, source port, destination address and destination port, so if a client changes its IP address or port number, any active TCP connection is no longer valid. Therefore, it needs the 3-way handshake to re-establish the connection. On the other hand, when QUIC client changes IP address or port number, it can continue the connection using the old connection ID, without re-handshaking or interrupting the connection. Indeed, this feature will be helpful in mobile devices during roaming.

4 Test-bed design and implementation

In this section, the test-bed design, implementation and tools used to perform the experiments and evaluation of QUIC is described.

4.1 Test-bed design

The goal of this section is to describe the creation of a test-bed network infrastructure where the various network impairment parameters such as packet loss, bit errors, and bandwidth limitations, *etc.*, can be controlled precisely. To achieve that, a test-bed shown in Fig. 4 is proposed which consists of two computers, one acting as a server that contains QUIC and TCP server code and the files to be downloaded. The other one acts as a client with QUIC and TCP client code installed on it. This setup scenario is designed to allow testing and comparing the two protocols. The two computers are networked using two scenarios: The first one is the controlled medium

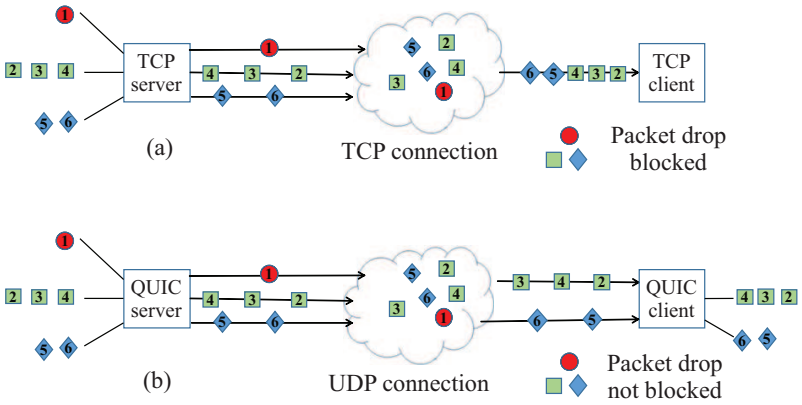

Fig. 3. (a) – Head of line blocking in TCP, (b) – Avoid head of line blocking in QUIC

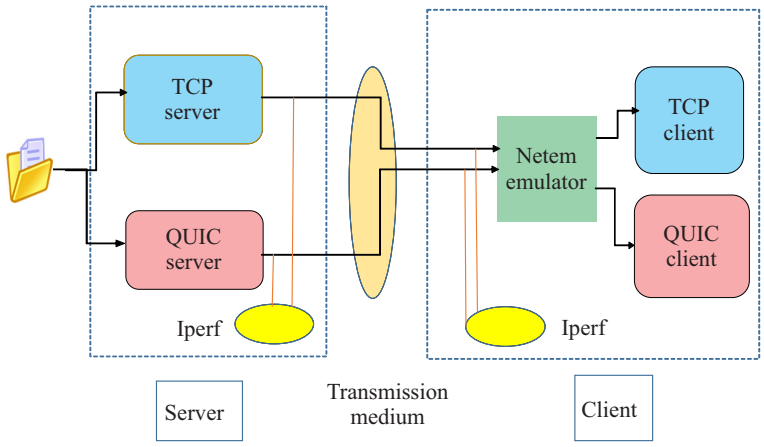

Fig. 4. Test-bed setup used for the controlled network experiments

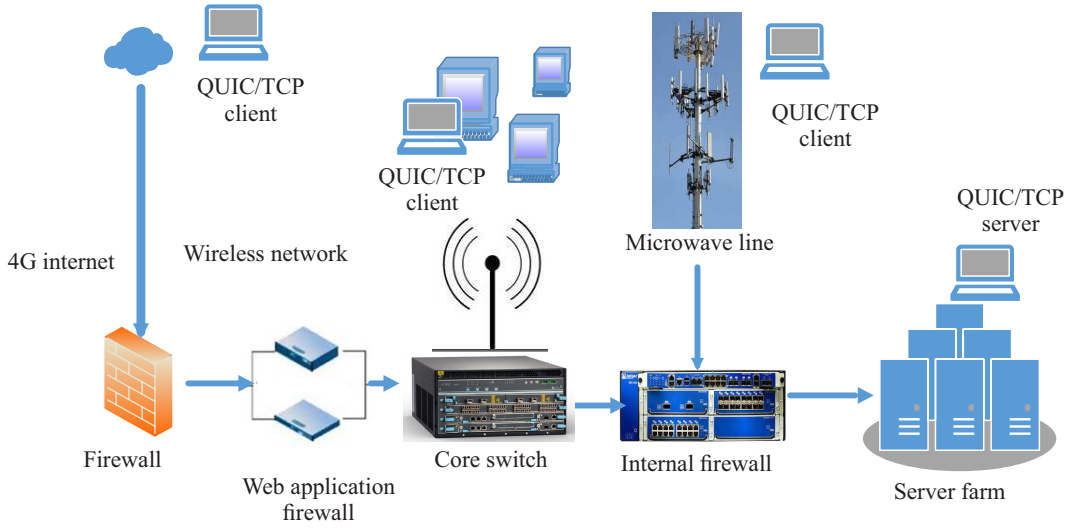

Fig. 5. Real network diagram used in the uncontrolled scenario

network, where computers are networked using a standard Ethernet cable to form a local area connection that is free of any network impairments such as delay, jitter, packet loss, *etc.* However, in order to study the effect of these parameters in a controlled manner, a network emulator tool such as NetEm [15] is installed on the client side, where different controlled network conditions can be configured. In addition, a traffic generator tool (IPerf) is used to generate background traffic in some tests in both computers[16]. The second scenario deals with an uncontrolled network depicted on Fig. 5, where real network mediums like leased lines, wireless and 4G Internet connections are used for testing and comparison between the two protocols with all security levels, in a production network devices such as switches, firewalls, web application filters, *etc.*

In this work, the measurement of the time taken by file transmission using QUIC and TCP protocols, with respect to different network conditions is recorded, for different controlled scenario as follows: the network emulator is used to generate an artificial packet loss by applying periodic packet losses of 2, 5, 10, 15 and 20%. Then, the test is repeated using Gilbert-Elliott loss model, which is a more realistic model that captures the temporarily correlation of lossy networks. This model will be discussed later in more details. After that, the wireless network behavior represented by bit errors is emulated using different bit errors values that range from 1 - 15%. Then, the effect of bandwidth limitation and background traffic, on the downloaded file is investigated by generating a background traffic using Iperf tool in both computers, while limiting the bandwidth using several values of 10, 20, 30 Mbps. In the uncontrolled scenario, different real network technologies are used such as: a microwave leased line with 2 Mbps bandwidth, Wi-Fi network, and a 4G Internet connection. Two cases have been conducted. The first one downloads a file of size 8.9 MB using either QUIC or TCP. This case is called unshared medium, while the second one downloads the same file simultaneously using the two protocols, this case is called shared

medium. In both cases, the experiment is repeated several times to account for network variations, and the average download time is recorded.

4.2 Test-bed setup and configuration

As shown in Fig. 4, the test-bed comprises several hardware and software components, including one computer acting as a server for both QUIC and TCP, another one acting as a client for QUIC and TCP network connection, NetEm, and Iperf software tools. Fig. 5 shows how to use the test-bed under different networking technologies. In what follows, a brief description of the different network technologies are used in the conducted experiments. In controlled network, a standard Ethernet cable link is used between the client and server with a speed of 100 Mbps between the two machines. On the other hand, the uncontrolled network shown in Fig. 5 utilizes several networking technologies such as: a microwave leased line with 2 Mbps speed, wireless local area network, and 4G wireless connection.

4.3 Client-Server Architecture

The test-bed consists of two HP pro-book 64-bit machines (server/client) running Ubuntu 14.04 with a Linux kernel 3.14.2. The QUIC server-client software installation is the main challenge of this test-bed; it is developed as a library inside the Chromium project's repository [17], not as a separate project. The QUIC library is currently built on C++ and includes Crypto and congestion control, accompanied with QUIC test server. Further, in order to compare QUIC performance with TCP, the Nginx software is installed as a web server [18] and used to download the file using TCP. The installation procedure steps are summarized as follows:

Step 1: create a QUIC folder, where the entire source coded is downloaded. Then, the depot_tools are checked out and installed into the QUIC folder using the following command:
$ git clone https://chromium.googlesource.com/chromium/tools/depot_tools.git

Step 2: fetch the chromium code by running the following command:
$ fetch –nohooks chromium

Step 3: install additional build dependencies by running the following command:
$./build/install-build-deps.sh

Step 4: download additional binaries by running the following command:
$ gclient runhooks

Step 5: update an existing checkout periodically by running the following command:
$ git rebase-update and: $ gclient sync

Step 6: Then, after these steps run without any errors, from the src directory, the following command should be executed to compile the QUIC server and client code:
$ ./build/install-build-deps.sh

Step 7: build the binaries using the following command
$ ninja -C out/Debug quic_server quic_client_epoll

The QUIC server and client binaries can be found in the src/out/Debug directory within Chromium directory. At this point, the server are ready to start running.

4.4 NetEm Tool

NetEm is a network emulator that changes packets' flow to mimic the behavior of real-networks [15], it offers functionalities for testing protocols by emulating the properties of a real network. The used version is (2.6) which emulates delay, packets' loss, duplication and reordering for a selected network interface. NetEm is used at the QUIC and TCP client side to emulate the various network impairments.

4.5 Iperf Tool

Iperf is a network tool developed to measure, diagnose, generate and test either TCP or UDP traffic throughput in IP networks. By setting various characteristics of TCP and UDP, the user can perform tests that declare the status of the network in the area of bandwidth availability, delay, jitter, and packet loss. Iperf is a command line program written in C language and it is an open source software with client and server functionalities. It runs on different platforms including Linux, UNIX, and Windows [16].

4.6 Transmission Medium

Different network technologies are used in the conducted experiments. In controlled network, a standard Ethernet cable link is used between the client and server with a speed of 100 Mbps between the two machines. On the other hand, the uncontrolled network shown in Fig. 5 utilizes several networking technologies such as: a microwave leased line with 2 Mbps speed, wireless local area network, and 4G wireless connection.

4.7 Practical considerations

In addition to the test-bed setup and configuration details described before, a number of practical considerations are taken into account to make the tests operate optimally. For example, the server and client operating systems must be 64 bits. The header of the files transferred through the network must be changed to enable transferring it via QUIC. This is done by a special editor program that does not corrupt the file like Emacs text editor, which is used to append the pre-header in the beginning of the file. In addition, QUIC code must be updated repeatedly to have an up-to-date code without bugs. Furthermore, QUIC depends on a certificate installed on both client and server; it must be renewed repeatedly to have the test-bed work perfectly. Finally, shell script is used to automate the conducted experiments and repeat it many times for more accurate results.

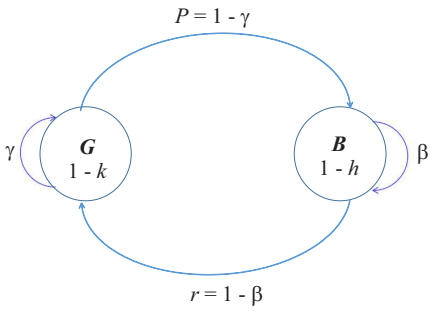

Fig. 6. Gilbert-Elliott modelling the temporal correlation of the lossy network[22]

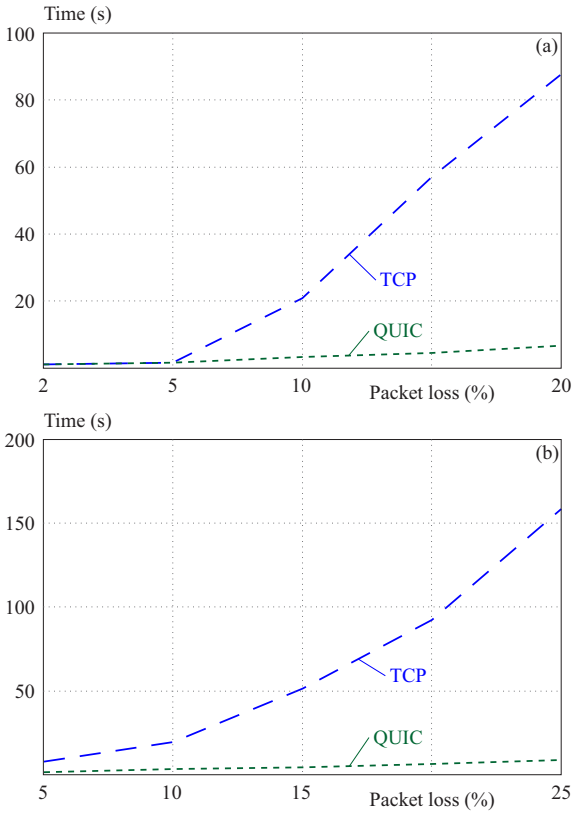

Fig. 7. Packet loss using (a) – Bernoulli, and (b) – Gilbert-Elliott models

5 Experimental results and performance evaluation

In this section, the experimental results acquired by running the two scenarios, controlled and uncontrolled networks clarified in Section 4 are presented. The first goal is to measure and compare the performance between QUIC and TCP in the two scenarios, the second one is to validate the seamless deployment of QUIC in real production networks without any difficulties. In the controlled network scenario, the network parameters are adjusted individually and their impact on QUIC and TCP performance are assessed experimentally as a function of download time. In the uncontrolled scenario, the network parameters vary depending on many surrounded circumstances such as users activities, speed variation, network congestion status and traffic conditions. Considering the fact that it is a real active network, network parameters are not predicted or known such as delay, jitter, packet loss, *etc.* Therefore, a large number of experiments for both QUIC and TCP are conducted, then the average file download time of these experiments is calculated. In what follows, the experimental results of the controlled and uncontrolled network scenarios are presented.

5.1 Controlled network scenario results

In this scenario, the following network parameters are adjusted and the results are shown in the figures attached for each condition:

5.1.1 Packet loss

NetEm tool is used to generate packet loss of 5–20 % in a step of 5 %, the downloaded file size is 8.9 MB and the available bandwidth is 100 Mbps. The experiment is performed in two packet loss models: Bernoulli or Independent model[19], and Gilbert-Elliott Model (GE)[20, 21]. The Bernoulli model among the most widely used models to resemble losses in the Internet, where packet loss or bit errors occur independently with a probability equals to M. For large number of packets N, the expected number of lost packets is NM. As Bernoulli model does not capture the actual behavior of network packet loss, through which packets loss happens and lasts for a time duration, the Gilbert-Elliott is a better choice to capture the temporal loss correlation of the Internet. Fig. 6 shows the GE model, which represents the network status using two states; Good (G) or state 1 and Bad (B) or state 0. The network starts transmission between these states. Each of these states may generate errors as independent events such as the error rate equals to $1-k$ and $1-h$ in the Good and the Bad states, respectively. According to state transition probabilities P and r, state 0 is a low loss state within dependent loss probability $P0$, while state 1 is a loss state with high independent loss probability $P1$

In Fig. 6, γ and β are the probabilities of self-transitioning for state G and B states, respectively. The overall probability of receiving q bits/packets from t - transmitted bits/packets under the GE model is calculated as,[21].

$$ploss = (1-k)\pi G + (1-h)\pi B \tag{1}$$

$$\pi B = p/(p+r), \quad \pi G = r/(p+r) \tag{2}$$

where πG, πB are the stationary state probabilities in state G, B, respectively, where $\pi G \gg \pi B$. GE Markov chain is used to capture temporally correlated pattern of packet loss. For the GE Markov chain, we apply transitioning probabilities of $\gamma = 0.99875$ and $\beta = 0.875$[21], then p, r, G and B are calculated as follows

$$p = 1 - \gamma = 1 - 0.99875 = 0.00125$$

$$r = 1 - \beta = 1 - 0.875 = 0.125$$

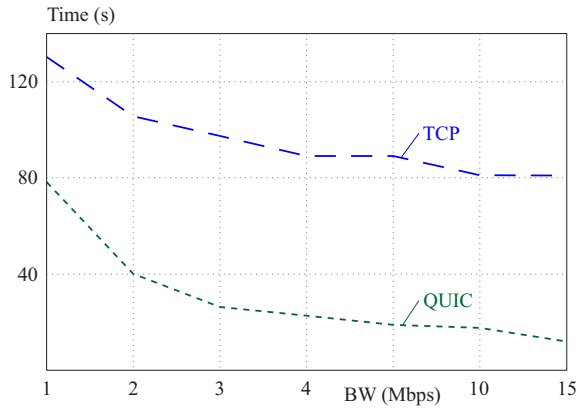

Fig. 8. Bandwidth limitation effect on TCP and QUIC

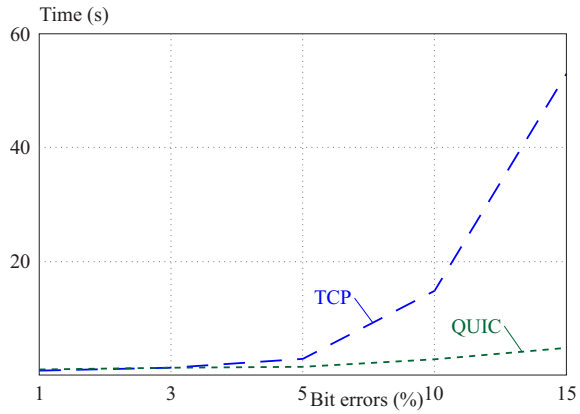

Fig. 9. Bit errors test on TCP and QUIC

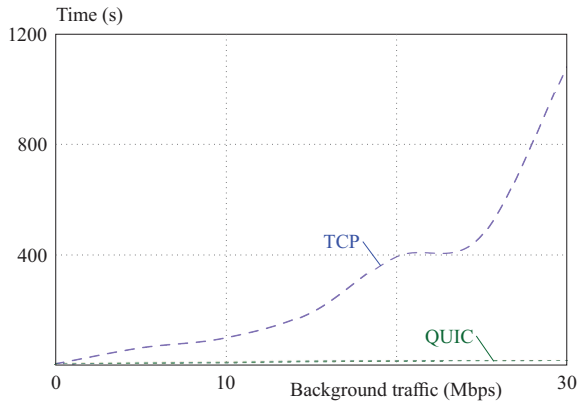

Fig. 10. Background traffic test on TCP and QUIC

$$\pi G = r/(p+r) = 0.125/(0.00125+0.125) = 0.99$$
$$\pi B = p/(p+r) = 0.00125/(0.00125+0.125) = 0.0099$$

Then $(1,2)$ are used to calculate the GE parameters that correspond to the packet loss under investigation. In this paper, five loss rates are tested (5, 10, 15, 20, and 25%). Let $h = 0.5$ then k can be estimated for all loss rates. For example, for $ploss = 0.05$, then k can be calculated as

$$ploss = 0.05 = (1-k)\pi G + (1-h)\pi B$$
$$1-k = 0.045 \text{ and } k = 0.955$$

After calculating the transition probabilities correspond to the above loss rates, the GE model is used to emulate packet loss using NetEm tool. As shown in Fig. 7 (a, b). QUIC performance is more efficient than TCP in both error models, especially at high packet loss. The reason in this case as discussed in Section 3, is that TCP suffers from HOLB in multiplexing, so all the streams are blocked waiting for the retransmission of lost packets, while QUIC surpasses this issue by limiting the impact only to the individual stream, and the other streams will continue to be transmitted.

5.1.2 Bandwidth limitation

Figure 8 compares the performance between QUIC and TCP when the bandwidth changes between 1-15 Mbps. The experimental results show that QUIC performance is significantly better than TCP. In this case, the average file download time of QUIC is approximately three times better than TCP. The main reason of this behavior is the 0-RTT connection establishment time of the QUIC, which proves to work efficiently.

5.1.3 Bit errors in wireless medium

Bit errors in wireless medium is also tested and the results are shown in Fig. 9, the range used for bit errors is 1–15 %. QUIC and TCP are comparable when the bit errors percentage is under 5%, but TCP shows significantly worse performance than QUIC as bit errors increase. The reason of this can be explained as the pacing in QUIC decreases the number of bits that have errors, especially at high bit error rates. Besides, the multiplexing in QUIC is preventing HOLB that TCP suffers from.

5.1.4 Network congestion emulation

In the controlled network with a bandwidth limit to 30 Mbps, a constant background UDP traffic varied from[0 - 30] Mbps is applied using Iperf tool to emulate congestion, as shown in Fig. 10, QUIC appears to be very suitable for this type of network when the bandwidth is limited, and appears to be very effective when the bandwidth is fully utilized.

The behavior in this case can be explained as QUIC uses the pacing congestion algorithm to prevent congestion as described in Fig. 3. In addition to packets reordering flexibility provided by QUIC protocol.

5.2 Uncontrolled scenario

In this scenario, the following network medium is used to evaluate QUIC performance and to compare it with TCP in real production networks. Three different networks are tested; a microwave leased line, WiFi and 4G Internet connections.

5.2.1 Microwave leased line

The test of this uncontrolled medium requires to put the test-bed server and client in different locations, and to use ordinary complex networks with many components such as switches, firewall and a microwave link over Multi protocol Label Switching (MPLS) with 2 Mbps speed. The experiment here considers the two cases described earlier; the unshared medium where a file of 8.9 MB size is downloaded using either QUIC or TCP. While the second case (the shared medium) downloads the same file simultaneously using the two protocols. In both cases, the experiments are repeated more than 20 times and the average download time is recorded when using QUIC and TCP for each case. The conducted experiments reveal several important observations and results. The first point to be confirmed from these experiments is that QUIC works smoothly like TCP without any additional configurations, in any network devices or intermediate nodes. The second point is that as shown in Fig. 11 (a), when the experiment is conducted in the unshared medium, the results show comparable performance value, but when the experiments are conducted in a shared medium, QUIC shows noticeable better performance as shown in Fig. 11 (b). Further, it is worth mentioning that the results match with what was explained in the controlled test, when applying background traffic due to the pacing congestion algorithm implemented by QUIC, which is used to prevent congestion, In addition, QUIC has more flexible reordering feature than TCP.

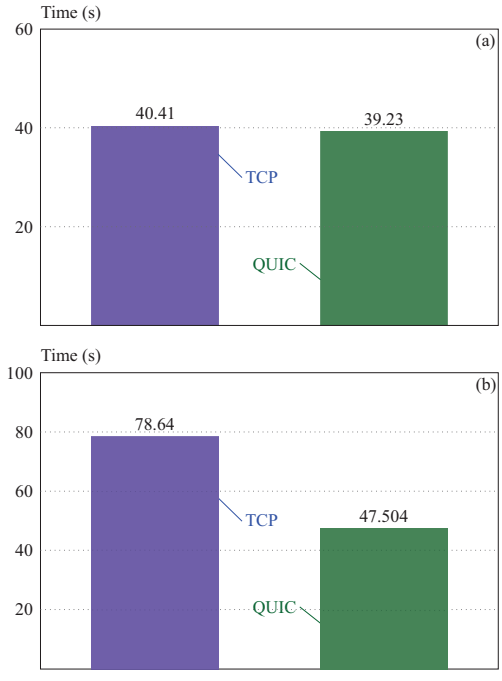

Fig. 11. Average file download time using TCP and QUIC tested under two cases (a) – unshared, and (b) – shared mediums

5.2.2 Wireless medium

As shown in Fig. 12, the results show that QUIC outperforms TCP; this is because normally WiFi networks are used for indoor coverage, which may have low bit errors rates. However, as examined in the controlled network scenarios, if the error rate increases, then QUIC significantly outperforms TCP.

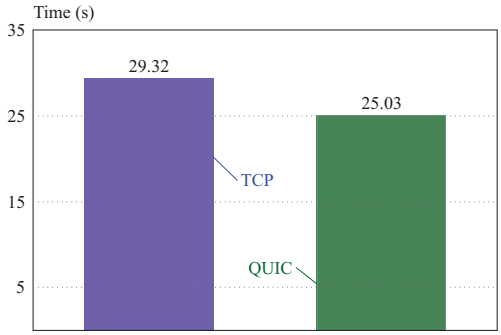

Fig. 12. Wireless medium test in QUIC and TCP

5.2.3 Internet 4G through SSL-VPN

To test the 4G Internet technology, the connection is established between the client and server through SSL-VPN. This connection does not require the installation of specialized client software on the end user's computer. It is used to give remote Internet users with access to web, client/server applications and internal network connections. The test takes place by measuring the average file download time, the experiment is repeated more than 10 times to account for network variations. As depicted in Fig. 13, QUIC outperforms TCP when using 4G Internet connection.

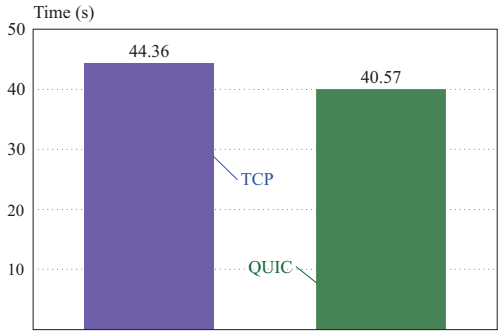

Fig. 13. 4G Internet test for QUIC and TCP

6 Conclusions and future work

In this paper, an experimental evaluation study has been conducted to evaluate the effectiveness of QUIC protocol in terms of average file download time, when compared with TCP over wireless networks. To achieve that, a test-bed that implements the QUIC server and client side software packages are installed. At the server

side, a network emulator is used to emulate the real network behavior in terms of packet loss, bit errors and bandwidth limitation in a controlled and deterministic manner. Furthermore, in order to assess the suitability and effectiveness of QUIC in real production networks, a series of experiments are conducted using three different wireless network technologies (microwave leased line, WiFi and 4G Internet connections), while downloading a file using several settings and background traffic. The conducted experiments have proven the effectiveness of QUIC protocol and showed its potential adoption as a substitution of TCP without changing or modifying the intermediate network infrastructure. As a future work, more experiments and tests will be conducted in real production networks, under different network impartment, and in different geographical locations. Furthermore, we plan to continue evaluating this protocol and expanding the testing to mobile devices.

References

[1] A. Khalifeh, M. Al-Taee, and A. Murshed, *Multimedia Tools and Applications*, Network-status aware quality adaptation algorithm for improving real-time video streaming over the internet, vol. 76, no. 24, pp. 26129–26152, 2016, DOI: 10.1007/s11042-016-3999-5.

[2] K. Darabkh, A. Awad, and A. Khalifeh, "New video discarding policies for improving UDP performance over wired/wireless networks", *International Journal of Network Management*, vol. 25, no. 3, pp. 181–202, 2015, DOI: 10.1002/nem.1888.

[3] K. Darabkh, A. Awad, and A. Khalifeh, "Efficient PFD-Based Networking and Buffering Models for Improving Video Quality over Congested Links", *Wireless Personal Communications*, vol. 79, no. 1, pp. 293-320, 2014, DOI: 10.1007/s11277-014-1857-1.

[4] A. Khalifeh and H. Yousefi'zadeh, "Optimal Audio Transmission Over Error-Prone Wireless Links", *IEEE Transactions on Multimedia*, vol. 12, no. 3, pp. 204-214, 2010, DOI: 10.1109/tmm.2010.2041096.

[5] A. Khalifeh and H. YousefiZadeh, "A Hybrid media scheme for wireless VoIP", *IEEE Data Compression Conference* (DCC) 2010.

[6] D. Somak, *Evaluation of QUIC on web page performance*, MS, Massachusetts Institute of Technology, USA, 2014.

[7] L. Gaetano, L. Cicco, and L. Mascolo, "HTTP over UDP: an experimental investigation of QUIC", *ACM Symposium on Applied Computing*, 2015.

[8] S. Albasrawi, "Performance analysis of Googles Quick UDP Internet Connection Protocol under Software Simulator", *Journal of Physics: Conference Series*, vol. 1591, p. 012026, 2020, DOI: 10.1088/1742-6596/1591/1/012026.

[9] Y. Gu and R. Grossman, "SABUL: A Transport Protocol for Grid Computing", *Journal of Grid Computing*, vol. 1, no. 4, pp. 377-386, 2003, DOI: 10.1023/b:grid.0000037553.18581.3b.

[10] Thomas, L., Dubois, and E., "Google QUIC performance over a public SATCOM access", *International Journal of Satellite Communications and Networking*, vol. 37, no. 6, pp. 601-611, 2019.

[11] Y. Cui, T. Li, C. Liu, X. Wang, and M. Kuhlewind, "Innovating Transport with QUIC: Design Approaches and Research Challenges", *IEEE Internet Computing*, vol. 21, no. 2, pp. 72-76, 2017, DOI: 10.1109/mic.2017.44.

[12] "SPDY: An experimental protocol for a faster web - The Chromium Projects", *Chromium. org*, 2021, https://www.chromium.org/spdy/spdy-whitepaper. Accessed: 10-Jan-2021.

[13] "QUIC: Design Document and Specification Rationale", *Docs. google.com*, 2021, https://docs.google.com/document/d/1RNHkx _VvKWyWg6Lr8SZ-saqsQx7rFV-ev2jRFUoVD34/mobilebasic [Accessed: 10- Jan- 2021].

[14] H. Bullot, R. Les Cottrell and R. Hughes-Jones, "Evaluation of Advanced TCP Stacks on Fast Long-Distance Production Networks", *Journal of Grid Computing*, vol. 1, no. 4, pp. 345-359, 2003, DOI: 10.1023/b: grid.0000037555.53402.4f.

[15] "Use Linux Traffic Control as impairment node in a test environment (part 2)", Excentis, 2021, https://www.excentis.com/blog/use-linux-traffic-control-impairment-node-test-environment-part-2 [Accessed: 10-Jan-2021].

[16] V. Guenat, "iPerf - iPerf3 and iPerf2 user documentation", Iperf.fr, 2021, https://iperf.fr/iperf-doc.php [Accessed: 10-Jan-2021].

[17] "Playing with QUIC - The Chromium Projects", Chromium.org, 2021, https://www.chromium.org/quic/playing-with-quic [Accessed: 10-Jan-2021].

[18] "The Architecture of Open Source Applications (Volume 2): nginx", Aosabook. org., 2021, http://www.aosabook.org/en/nginx.html [Accessed: 10-Jan-2021].

[19] P. McCullagh and J. Nelder, *Generalized linear models*, Boca Raton: Chapman & Hall, 1999.

[20] H. Yousefi'zadeh, H. Jafarkhani and M. Moshfeghi, "Power Optimization of Wireless Media Systems With Space-Time Block Codes", *IEEE Transactions on Image Processing*, vol. 13, no. 7, pp. 873-884, 2004, DOI: 10.1109/tip.2004.827234.

[21] G. Hablinger and O. Hohlfeld, "The Gilbert-Elliott model for packet loss in real time services on the Internet, Measuring", *Modelling and Evaluation of Computer and Communication Systems*, (MMB)", in 14th GI/ITG Conference, 2008.

Received 30 October 2020

Ala' Khalifeh received the PhD degree in Electrical and Computer Engineering from the University of California, Irvine -USA in 2010. He is currently an associate professor in the the Electrical Engineering department at the German Jordanian University. He is currently the IEEE Jordan section chair. His research is in communications technology, and networking with particular emphasis on optimal resource allocations for multimedia transmission over wired and wireless networks, Internet of Things and wireless sensor networks.

Ma'moun Mansour was born in 1978, he received the MSc degree in Computer Engineering from the German Jordanian University, Jordan in 2017. He is currently a section header of network and systems in Information Technology department at Greater Amman Municipality, and responsible for new network and systems projects. His research is in network protocols. This is the first publish research for him.

Sahel Alouneh is full professor of computer engineering at Al Ain University. He held the post of Dean of the Faculty of electrical engineering and information technology and the Dean of Scientific Research at the German Jordanian University. His research interests include computer and communication networks, big data security, cloud computing, software security, MPLS security and recovery, wireless networks security, software testing, computer design, and architecture.