Effect of Path QoS on Throughput Aggregation Capability of the MPT Network Layer Multipath Communication Library

Naseer Al-Imareen, and Gábor Lencse

Abstract—**An increase in the use of smart and portable devices like smartphones, laptops, and tablets has led to a rise in the number of network interfaces and thus the number of possible channels for communication. However, the current approach over the Internet only employs a single path for a communication session. As an innovative and promising method for real-time transmission based on GRE-in-UDP encapsulation, which provides an IPv4 or IPv6 tunneling mechanism, this paper presents multipath throughput testing for the MPT network layer multipath communication library. We investigated the effectiveness of MPT's channel capacity aggregation while dealing with wired channels and examined scenarios in symmetric and asymmetric paths. Our network throughput measurements showed that MPT can efficiently aggregate the capacities of both symmetric and asymmetric paths. In this paper, we established a network topology that included a server, which we used for generating various quality of service (QoS) metrics. We measured how latency, transmission speed, packet loss rate, jitter, and the setting of the path weights influence throughput aggregation capability of the MPT communication library.**

Index Terms—**GRE-in-UDP, MPT, MPTCP, Tunneling, Throughput, QoS.**

I. Introduction

Many factors contribute to the popularity of multipath approaches. We use many IT gadgets daily (smartphones, tablets, laptops) with more than one network interface (Wi-Fi, 4G/5G, Ethernet). However, the TCP/IP protocol stack was designed to support only a single interface per communication session. Thus, the current environment of the Internet allows only a single path to transfer packets in a communication session. The single-path connection technologies cannot use the advantages of multiple interfaces. This calls for the creation of multipath-friendly infrastructure and protocols for use in the communication session. Multipath communication has emerged as a hot research topic in recent years. When multiple interfaces are allowed in a single communication session, throughput increases dramatically and generally. The multipath technique has several promising solutions, such as MPT [1] and MPTCP [2]. MPT can be a key technology for making the most of the various connection points offered by today's electronic gadgets for exchanging information. MPT operates at the network layer via the GRE-in-UDP encapsulation. In contrast, MPTCP implements the TCP protocol at the transport layer [3]. Using the network layer for multipath communication is the focus of our paper. The 32-bit and 64-bit versions of MPT are available for free download from [4].

This paper is an extension of our former conference paper [5] in which we studied the efficiency of channel capacity aggregation of MPT using a single physical switch. In this paper, we use a more complex test network to measure how latency, transmission speed, packet loss rate, and jitter affect the throughput aggregation of MPT. The main contribution of our paper is analyzing how the various path quality of service (QoS) metrics, and the setting of the path's weights, affect the throughput aggregation capability of the MPT network layer communication library using both symmetric and asymmetric paths.

The rest of this paper is organized as follows. In section II, we give a brief introduction to MPT. In the third section, we provide an overview of the related work. The fourth section explains our experimental test environments in both hardware and software configuration. The fifth section includes the various MPT measurements and results. Finally, the sixth section is a discussion and future directions of research, and this is followed by the conclusion.

II. Brief Overview of MPT

MPT implements multipath communication in the network layer. It was developed and designed by a research group at the University of Debrecen, Hungary [1]. MPT takes advantage of GRE-in-UDP capabilities to provide multipath tunneling. MPT can act as a router through which packets can be routed between different networks through tunnel endpoints. This feature is the establishment of a multipath connection from one site to another. The IP version of the tunnel is independent of the IP version of the path so that the MPT can be used for IPv6 transition purposes [6], [7] and [8].

The MPT library creates a tunnel interface to serve as a logical interface for communication between hosts. The logical interface is mapped to the physical interfaces directly by MPT software [9][10]. When an application sends an IP

Naseer Al-Imareen, is with the Department of Telecommunications, Széchenyi István University, Győr, Hungary; Computer Science, Al-Qadisiyah University, Iraq (e-mail: al-imareen.naseer@sze.hu).

Gabor Lencse is with the Department of Telecommunications, Széchenyi István University, Győr, Hungary (e-mail: lencse@sze.hu).

DOI: 10.36244/ICJ.2023.2.3

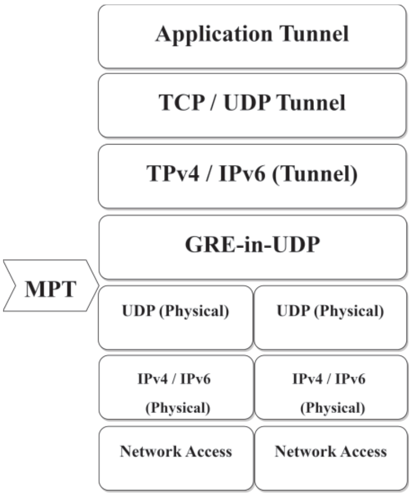

Fig. 1. The Conceptual Architecture of MPT-GRE [1].

packet, it uses the address of the tunnel path, i.e., the address of the logical interface. Thus, there is no need to modify the applications.

The fact that the MPT library enables communication through the tunnel interface helps in mapping between the tunnel interface and physical interfaces. For example, when changing the physical interface address or the physical interface itself, the application does not know about this change. It continues to work through the tunnel interface where the MPT library reorganizes the mapping from the logical interface to the physical interface based on the latest update. Transmission of video in high quality in real-time is one of the challenges that can be solved using multipath technology. The developers and researchers of solutions have proposed to produce and develop technologies for the multipath problem that are essential in facing this challenge.

The MPT library provides a typical solution for multipath by giving a logical path (tunnel) that is initialed in the terminal hosts to define the socket where the MPT library reads the packets coming at the logical interface (IPv4 or IPv6) originating from the sender host. This packet will be wrapped in a new GRE on the UDP part before being sent over a potential physical route [11]. The MPT-GRE conceptual architecture can be seen in Fig. 1. This figure displays the MPT's layered design. MPT expands upon the GRE-in-UDP design by permitting multiple physical channels [12]. MPT is comparable to MPTCP in this regard, but unlike MPTCP, it employs UDP in the underlying layer, builds on GRE-in-UDP, and provides a tunnel IP layer over which both UDP and TCP can be used.

III. LITERATURE REVIEW

The aggregation capability of the MPT network layer communication library is among the main challenges in different fields that help examine the efficiency of multipath technology. Almási and Szilágyi [13] proved the MPT library's effective throughput aggregation property in IPv6 and mixed (i.e., protocol version transition) contexts. The main goal was to check how well the MPT tool handled throughput in IPv4, IPv6, and hybrid network environments. They set up a measurement system with four connection pathways for communication hosts. Different data transmission sizes and symmetrical and asymmetrical bandwidth rates were used to test the throughput performance. The test results showed that the MPT multipath environment correctly adds up the throughput capacity of each physical path, even though the results from the dissimilar protocol versions were different.

Kovács [14] tested the ability to throughput aggregate an MPT communication library and compared it with multipath in TCP up to 12 connections with all possible combinations of IPv4 and IPv6. He found that throughput scaled up linearly. The researcher also found the possibility of using 12 NICs using the MPT communications library at a speed of 100Mbps on relatively old computers. The same author has expanded his measurements to 1 Gbps speed links, and he has also prepared a mathematical model for the throughput aggregation capability of MPT and MPTCP in [15].

Using multiple scenarios carried out on quad-path Gigabit Ethernet and IPv4/IPv6 connections, the authors of [8] compared the MPT solution to the MPTCP, which was used as a benchmark. The primary benefit and ideal use of the MPT is UDP multimedia transport. However, the solution's throughput performance is more than sufficient.

Szilágyi et al. [16] compared two technologies MPTCP and MPT that support multipath communication sessions. The researchers performed an efficiency analysis of the 10 Gigabit Ethernet dual-path scenario for two techniques providing multipath connectivity. This could be helpful in data center environments by increasing efficiency and enhancing user experience in the cloud. Also, these multipath technologies can improve Gigabit Ethernet throughput in the data center.

The authors of [17] addressed guaranteeing the QoS of transmission of Network-based solutions for Multipath TCP. By using the advantage of Software-defined networking technology, they created a method that can allocate the available paths deterministically by combining network sites from the network's perspective, metrics like link bandwidth and latency can be gathered to assess the overall health of the system. By comparing the needed and the available resources in the controller, the endpoint will be given the optimum number of subflows. Their improved method addresses some multithreading issues and has been shown to increase network throughput. To lower the latency of the client-server connection, Google recently developed the Quick UDP Internet Connection (QUIC) protocol, which integrates the features of HTTP/2, TLS, and TCP directly via UDP [18]. The authors [19] suggested a new technique for measuring QUIC's passive delay that combines a delay bit and a spin bit. They have demonstrated that this new methodology can solve the drawbacks of measurements based solely on the spin bit. However, this brand-new delay-bit technique only adds one bit to the spin bit.

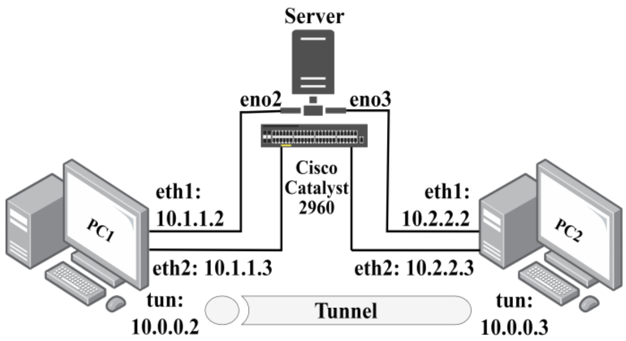

Fig. 2. Measurement Network Topology.

IV. Experimental Test Environment

A. Hardware and Fundamental Configurations

We have built a test system for measurements and analysis as shown in Fig 2. We used two Dell Precision Workstation computers with the following specification:

- Motherboard with Intel 5000X chipset.
- 8 GB 533MHz DDR2 SDRAM.
- Intel Xeon 5140 2.33 GHz dual core processors.
- NIC 1: Broadcom NetXtreme BCM5752 Gigabit Ethernet controller.
- NIC 2: Intel PT Quad 1000 type four-port Gigabit Ethernet controller. (Two ports were used for the experiments.)

We used a server to emulate various QoS issues and a Cisco Catalyst 2960 switch [20] to link the other two interfaces and limit the transmission speed to 100Mbps. The IP address configuration of our test network is also shown in Fig. 2. Two independent paths were established between the two hosts (eth1, eth2) and the logical interface (tun0) was provided by the MPT software. Debian 8.11 GNU / Linux operating system was installed on both computers, as this version is compatible with precompiled library files from 2019. The intent of this measurement network was to examine the throughput aggregation efficiency of the MPT software and to monitor and verify the effect of various QoS parameters.

B. MPT Software Configuration

The MPT implementation contains two versions (32-bit and 64-bit). In our paper, the mpt-gre-lib64-2019.tar.gz version was used. We have downloaded the MPT from [4]. We had to edit two configuration files (the paths are relative to the installation directory of MPT). The first file `conf/interface.conf` was:

```
; ####### General Interface Information #######
  60456         # The local cmd UDP port number
  2             # the interfaces number
  1             # Accept remote request
; MPT software make mptsrv as a server
  25            # cmd_timeout
; ############ Information tun0 ##############
  tun0          # Name of tunnel interface
  1440          # Maximum Transfer Unit
  10.0.0.2/24   # IPv4 address and prefix length
```

The same configuration method in the above file was used for the other interface on the second computer. Different types of tunnels files were placed in independent connection files. Moreover, all the information regarding IP addresses and a prefix length for each tunnel created by MPT software were prepared.

Additionally, the session connection file was prepared and configured in which the connection configurations IPv4 over IPv4 are defined in the `conf/connections /IPv4.conf` file:

```
; ####### Connections Info. ########
  1              # Connections Num.
; ####### Connection Details #######
MPT_Connection   # Connection Name
3                # Permission Send (1)/Receive (2)
4                # The Version of IP
10.0.0.2         # Local_IP
50230            # Local_Port
10.0.0.3         # Remote_IP
50230            # RDP (Remote_Data_Port)
60456            # UDP Port Number_Remote
2                # Paths_Num.
0                # Networks_Num
5                # Time of Keepalive Message
5                # Dead_Timer (sec)
0                # Status of Connection
0                # Auth_Type
0                # Auth_Key
; ########### Path0 info. ##########
eth1             # Interface Name
4                # The Version of IP
10.1.1.2/24      # Public IP
10.1.1.2         # Gateway_IP
10.1.1.3/24      # Remote IP
5                # Keepalive Time
25               # Dead Time
100              # Weight_out
100              # Weight_in
1                # CMD_default
; ####### Path1 information ########
eth2             # Interface Name
4                # IP_Version
10.2.2.2/24      # Public IP
10.2.2.2         # Gateway_IP
10.2.2.3/24      # Remote IP
5                # Keepalive Time
25               # Dead Time
100              # Weight_out
1                # Weight_in
1                # CMD_default
0                # Path status
```

The remaining paths for this connection were prepared and configured in the same manner that was adopted for the above two paths. It is worth noting that the configuration files must adhere to a tight structure, including comment-only lines. In [6], the researchers suggested that this be adjusted for the most widely used freestyle configuration files using keyword parsing.

V. MPT Measurements and Results

To examine and analyze the effect of path QoS on the throughput aggregation capability of the MPT network layer multipath communication library, more scenarios were implemented using the iPerf3 [21] real-time network performance measurement tool. It is a cross-platform, open-

TABLE I
THROUGHPUT OF ASYMMETRIC PATHS WITH DELAY APPLIED TO BOTH NETWORK INTERFACES OF THE SERVER

Delay / Interface	0ms	10ms	20ms	30ms	40ms	50ms
eth1	9.9	9.8	9.79	9.77	9.72	9.39
eth2	94.8	94.7	94.7	94.8	94.8	94.7
tun0	103.5	98.61	88.71	78.73	66.16	52.65

TABLE II
THROUGHPUT OF ASYMMETRIC PATHS WITH DELAY APPLIED TO ONE NETWORK INTERFACE OF THE SERVER

Delay / Interface	0ms	10ms	20ms	30ms	40ms	50ms
eth1	9.9	9.8	9.8	9.8	9.8	9.7
eth2	94.8	94.8	94.7	94.8	94.8	94.7
tun0	103.6	102	98.45	93.52	90.55	82.99

source client-server program that may be used to test the throughput between two hosts. We used the following style commands on the client side:

```
iperf3 -c 10.2.2.2 -t 30 -f M
```

This command ran a 30-sec experiment and returned the throughput in MB/sec units. The server, on the other hand, was launched with the following command line:

```
iperf3 -s -f M
```

We applied the additional server to be able to adjust various QoS metrics of the bypassing traffic using the *tc* tool of Linux. We experimented with different values of delay, packet loss, and different speeds. The MPT server program allocates the outgoing packets among the network paths according to their weights; in the asymmetric paths, we assigned each path a weight equal to its transmission speed except for the last experiment when we tested the effect of proper and improper weight settings.

A. Experiments with Network Delay

We generated various delays, where the *x* millisecond delay was set to 10ms, 20ms, 30ms, 40ms and 50ms.

```
tc qdisc add dev eth1 root netem delay xms
```

Table I shows the effect of delay on the throughput where the delay was added to both network interfaces of the server. (It means that the packets were delayed in both directions.) Analyzing the results shows that while the effect of the delay on the throughput of the path was minor (it was reduced from 9.9Mbps to 9.39Mbps), its effect on the tunnel was more significant, where the throughput reduced from 103.5Mbps (with no delay) to 52.65Mbps (with 50ms delay).

On the other hand, we added the delay on one of the two network interfaces of the server. The results in Table II show the effect of delay on the throughput. Analyzing the results shows a lower throughput effect on the delayed port (reduced from 9.9Mbps to 9.7Mbps) compared to the previous experiment when the delay was added to both ports. Additionally, the throughput effect on the tunnel (reduced from 103.6Mbps with no delay to 82.99Mbps with a 50ms delay) was also lower compared to the previous experiment.

TABLE IV
THROUGHPUT OF ASYMMETRIC PATHS WITH DIFFERENT PACKET LOSS RATES

Packet loss / Interface	0 %	1%	2%	3%	4%	5%
eth1	9.8	9.43	9.33	8.7	7.7	4.6
eth2	94.8	94.8	94.8	94.8	94.7	94.8
tun0	103.5	24.5	13.9	9.02	6.1	4.1

B. Experiments with Transmission Speed Limit

We applied various transmission speed limitations, where *x* was set to 10Mbps, 20Mbps, …, and 100Mbps using Token Buffer Filter (TBF) to influence traffic speed. The data transmission speed was controlled by applying rate-limiting instructions to both network interfaces of the server using the following command:

```
tc qdisc add dev eno2 root tbf rate x latency 400ms
```

We always allocated the path weight equal to its transmission speed. Table III shows the effect of using asymmetric paths with different transmission speeds. Analyzing the results shows that the value of tunnel throughput is very close to the sum of the path throughputs. We can conclude that the aggregation is very efficient if the weight of the paths is equal to their speeds. Please refer to Section V.E for our experiments using weights that are different from the path speed.

C. Experiments with Packet Loss

We tested various packet loss cases as follows: the percentage of lost data x was set to 1%, 2%, 3%, 4% and 5%. The packet loss metric was added to both network interfaces of the server using the following command:

```
tc qdisc add dev eno2 root netem loss x
```

Table IV shows the effect of packet loss on the throughput. Analyzing the results shows that the effect of the packet loss on the throughput of the individual paths was negligible. However, the effect on the throughput of the tunnel was clearly obvious, where the throughput reduced from 103.5 Mbps (with no packet loss) to 4.1 Mbps (with 5% packet loss). This caused serious performance issues for the tunnel in MPT communication.

TABLE III
THROUGHPUT OF ASYMMETRIC PATHS WITH DIFFERENT TRANSMISSION SPEEDS

Speed / Interface	10Mbps	20Mbps	30Mbps	40Mbps	50Mbps	60Mbps	70Mbps	80Mbps	90Mbps	100Mbps
eth1	9.84	19.9	29.9	39.9	49.8	59.7	69.5	78.9	88.6	94.8
eth2	94.7	94.7	94.7	94.7	94.7	94.7	94.7	94.7	94.7	94.8
tun0	103.4	112.7	121.9	132.2	140.3	149.6	159	168	177.2	186

TABLE V
JITTER OF SYMMETRIC AND ASYMMETRIC PATHS

Interfaces	Jitter (ms)	
	Symmetric	Asymmetric
eth1	0.104	0.132
eth2	0.111	0.118
tun0	0.179	0.158

D. *Experiments with Jitter*

We considered two successive packets that were sent to the traffic, $C_0$and C_1. For packet C_j , $j = 0{,}1$ and node $k = 1, \dots, n$, let $T_j^{in}(k)$ and $T_j^{out}(k)$ be the times of incoming and departure packets of C_j at node k, let $W_j(k)$ be the waiting time of C_j at node k, and finally let $\Delta_k = \big(W_1(k) - W_0(k)\big)$ be the variation of the inter-packet delay at node k. The jitter is defined as follows [22]:

$$J_{[1..n]}(T) = E\left[\ \left|\sum_{k=1}^{n}\big(W_1(k) - W_0(k)\big)\right|\ \right]=$$

$$= E\left[\left|\ \sum_{k=1}^{n}\Delta_k\ \right|\right] \qquad (1)$$

Consequently, the jitter is clarified as the predicted absolute value of the total packet delay fluctuations imposed by each node along the route from source to destination. As a lower jitter score indicates a more constant and dependable response time, it is safe to assume that a connection is good.

Higher jitter scores indicate more noticeable variations in reaction times [23], [24].

We tested various situations where 20 packets were sent during the second topology and in the case of symmetric and asymmetric paths from the source to the destination.

We calculated the round-trip times (RTT) of symmetric paths, representing the network latency that it takes for a request to go from the first node to the destination and back again. This time variation, expressed in milliseconds, affects the typical time of transmitting data packets when it is clear and large.

The jitter was calculated using equation (1). It was found that the variance in the jitter was good [25] because it was less than 30ms, as shown in Table V. The results proved the reliable operation of MPT multipath communication library in this situation because in both cases the jitter value was a small score.

E. *Experiments with Various Path Weights*

We have measured the throughput of the tunnel for different cases, where the path speeds were tested from 10Mbps to 100Mbps. For each path, we examined different weights from 10 to 100. The speed and weight of the second path was always kept constant at 100Mbps and 100, respectively (Fig. 3 shows the results). It needs to be noted that we also measured the throughputs of the individual paths. It was about 94.8Mbps for a 100Mbps link and 9.85Mbps for a 10Mbps link.

On the one hand, using proper weights (that is, the weight of the path is equal to the speed of the path) the throughput aggregation is nearly perfect. The throughput of the tunnel is close to the sum of the throughput of the paths. Let us see two extreme examples:

- When the first path had 100Mbps speed and 100 weight, the throughput was 186Mbps, nearly the sum of the throughputs of the two individual paths.
- When the first path had 10Mbps speed and 10 weight, the throughput was 103.4Mbps, nearly the sum of the throughput of the two individual paths.

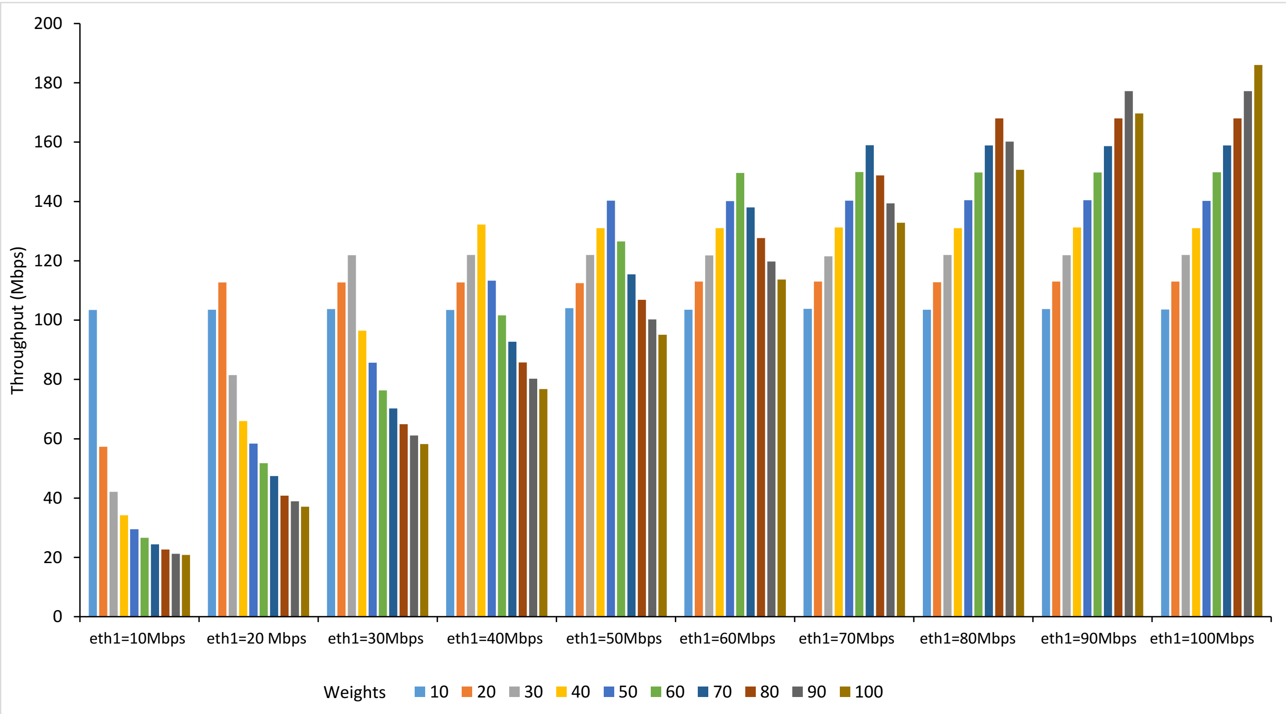

Fig. 3. The tunnel throughput as a function of different transmission speeds and weights for path 1.

On the other hand, any improper weight setting deteriorates the throughput aggregation capability of MPT. Let us see two extreme examples:

- When the first paths had 100Mbps speed and 10 weight, the throughput was 103.6Mbps, that is, the first path could contribute to the throughput only as if it were a 10Mbps path.
- When the first path had 10Mbps speed and 100 weight, the throughput was 20.8Mbps, that is, the second path could contribute to the throughput only as if it were also a 10Mbps path.

In other words, if equal weights are used for asymmetric paths, the faster link can contribute to the throughput as much as the slower link.

VI. CONCLUSION

We have tested the channel aggregation capability of the MPT library, which provides a network layer multipath solution based on GRE-in-UDP between two hosts with the possibility of using any transport layer protocol (TCP or UDP). Our test system enabled us to experiment by changing various QoS metrics (latency, transmission speed, packet loss rate, and jitter) and examine how their different values affect the throughput aggregation capability of the MPT network layer multipath communication library. We found that the distribution of the outgoing packets among the links according to their weights makes the throughput of the tunnel close to the sum of the throughput of the two paths. However, improper weights, as well as increasing delay and frame loss rate of one of the paths, may cause a significant deterioration of the throughput of the tunnel.

ACKNOWLEDGEMENT

The authors thank John Kowalchuk, Széchenyi István University, for the English language proofreading of the manuscript.

REFERENCES

[1] Almási, B., Lencse, G., and Szilágyi, S., "Investigating the multipath extension of the GRE in UDP technology," *Comput. Commun.*, vol. 103, pp. 29–38, 2017. DOI: 10.1016/j.comcom.2017.02.002

[2] Ford, A., Raiciu, C., Handley, M., and Bonaventure, O., "TCP extensions for multipath operation with multiple addresses", IETF, Jan. 2013, RFC 6824.

[3] Jabbar, N. and Lencse, G., "An Overview of the multipath technologies, their importance and types," *Proceedings of The Technical University of Sofia*, 72(1), pp. 21–28, 2022. DOI: 10.47978/TUS.2022.72.01.004

[4] Fejes, F., "MPT – multi-path tunnel", precompiled version can be downloaded from: http://github.com/spyff/mpt, 2019.

[5] Jabbar, N. and Lencse, G., "Measurement and analysis of MPT multipath throughput in wire channels," *2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, pp. 1–5, IEEE, 2022. DOI: 10.1109/ICECCME55909.2022.9987956

[6] Lencse, G. and Kovács, Á., "Advanced measurements of the aggregation capability of the MPT network layer multipath communication library," *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems*, vol. 4, no. 2, pp. 41–48, 2015. DOI: 10.11601/ijates.v4i2.112

[7] Szilágyi, S., Fejes, F., and Katona, R., "Throughput performance comparison of MPT-GRE and MPTCP in the Fast Ethernet IPv4/IPv6 environment," *Journal of Telecommunications and Information Technology*, vol. 2, pp. 53–59, 2018. DOI: 10.26636/jtit.2018.122817

[8] Szilágyi, S., Bordán, I., Harangi, L., and Kiss, B., "Throughput performance comparison of MPT-GRE and MPTCP in the gigabit Ethernet IPv4/IPv6 environment," *J. Electra. Electron.* Eng., vol. 12, no. 1, pp. 57–60, 2019.

[9] Lencse, G. and Kovács, Á., "Testing the channel aggregation capability of the MPT multipath communication library", *World Symposium on Computer Networks and Information Security 2014 (WSCNIS 2014)*, Hammamet, Tunisia, Paper ID: 1569946547, 2014.

[10] Gál, Z., Almási, B., Dabóczi, T., Vida, R., Oniga, S., Baran, S., & Farkas, I., "Internet of things: application areas and research results of the FIRST project", *Infocommunications Journal*, vol. 6, no. 3, pp. 37–44, 2014.

[11] Szilágyi, S. and Bordán, I., "Throughput performance measurement of the MPT-GRE multipath technology in emulated WAN Environment," *CEUR Workshop Proc.*, vol. 2874, pp. 187–195, 2021.

[12] Szilágyi, S. and Bordán, I., "Investigating the through put performance of the MPT-GRE network layer multipath library in emulated WAN environment," *Acta Technica Jaurinensis*, vol. 15, no. 1, 2022. DOI: 10.14513/actatechjaur.00639

[13] Almási, B. and Szilágyi, S., "Investigating the performance of the MPT multipath communication library in IPv4 and IPv6," *International Journal of Advances in Telecommunications, Electrotechnics, Signals and Systems*, vol. 5, no. 1, pp. 53–60, 2016. DOI: 10.11601/ijates.v5i1.148

[14] Kovács Á., "Comparing the aggregation capability of the MPT communications library and multipath TCP", *2016 7th IEEE International Conference on Cognitive Infocommunications (CogInfoCom)*, pp. 157–162, 2016. DOI: 10.1109/CogInfoCom.2016.7804542

[15] Kovács Á., "Evaluation of the Aggregation Capability of the MPT Network Layer Multipath Communication Library and Multipath TCP", *Acta Polytechnica Hungarica*, vol. 16, no. 6, pp. 129–147. DOI: 10.12700/APH.16.6.2019.6.9

[16] Szilágyi, S., Bordán, I., Harangi, L., and Kiss, B., "Throughput performance analysis of the multipath communication technologies for the cloud," *Journal of Electrical and Electronics Engineering*, vol. 12, no. 2, pp. 69–72, 2019

[17] Gao, K., Xu, C., Qin, J., Yang, S., Zhong, L., and Muntean, G., "QoS-driven path selection for MPTCP: A scalable SDN-assisted approach," *2019 IEEE Wireless Communications and Networking Conference (WCNC)* pp. 1–6, IEEE, 2019. DOI: 10.1109/WCNC.2019.8885585

[18] De Coninck, Q., and Bonaventure. O., "Multipath QUIC: Design and Evaluation," *In Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies (CoNEXT '17)*. Association for Computing Machinery, New York, NY, USA, pp. 160–166. DOI: 10.1145/3143361.3143370

[19] Bulgarella, F., Cociglio, M., Fioccola, G., Marchetto, G., and Sisto, R., "Performance measurements of QUIC communications," *In Proceedings of the Applied Networking Research Workshop (ANRW '19)*. Association for Computing Machinery, New York, NY, USA, pp. 8–14. DOI: 10.1145/3340301.3341127

[20] "Cisco Catalyst 2960 Series Switches." Cisco. URL https://www.cisco.com/c/en/us/support/switches/catalyst-2960-series-switches/series.html.

[21] Mortimer, M., "Iperf3 documentation," 2018. https://buildmedia.readthedocs.org/media/pdf/iperf3-python/latest/iperf3-python.pdf

[22] Brun, O., Bockstal, C., and Garcia, J. M., "A simple formula for end-to-end jitter estimation in packet-switching networks," *International Conference on Networking, International Conference on Systems and International Conference on Mobile Communications and Learning Technologies (ICNICONSMCL'06)*, pp. 14–19, IEEE, 2006. DOI: 10.1109/ICNICONSMCL.2006.34

[23] Castro, M., Márquez, L., and Márquez, J., "A comparative analysis of QoS parameters measurement applying packet scheduling algorithms on Cisco 2800 Router with Static Routing," *2020 6th International Conference on Science in Information Technology (ICSITech)*, pp. 23–28, IEEE, 2020. DOI: 10.1109/ICSITech49800.2020.9392032

[24] Al-Shaikhli, A., Esmailpour, A., and Nasser, N., "Quality of service interworking over heterogeneous networks in 5G," *2016 IEEE international conference on communications (ICC)*, pp. 1–6, IEEE, 2016. DOI: 10.1109/ICC.2016.7510913

[25] Armitage, G., and Stewart, L., "Limitations of using real-world, public servers to estimate jitter tolerance of first-person shooter games," *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in computer entertainment technology*, pp. 257–262, 2004. DOI: 10.1145/1067343.1067377

Naseer Al-Imareen (received his MSc in Computer Science at the Informatics Institute for Graduate Studies of Baghdad (Iraq) in 2019.
He has worked for the Department of computer science, Al-Qadisiyah University in Iraq since 2008. He taught computer networks, data structure, operating systems, and web design. Now, he is a PhD student at Széchenyi István university of Győr, Hungary. The area of his research includes the performance analysis of computer networks and investigation of quality of service.

Gábor Lencse received his M.Sc. and Ph.D. degrees in computer science from the Budapest University of Technology and Economics, Budapest, Hungary in 1994 and 2001, respectively.
He works for the Department of Telecommunications, Széchenyi István University, Győr, Hungary since 1997. He is now a professor. He is also a part time Fellow at the Department of Networked Senior Research Systems and Services, Budapest University of Technology and Economics since 2005. His research interests include the performance and security analysis of IPv6 transition technologies.