Adaptive Control of Congestion Window in QUIC

So-Yong Kim
School of Computer Science and Engineering
Kyungpook National University
Daegu, Korea
thdyd324@gmail.com

Seok-Joo Koh
School of Computer Science and Engineering
Kyungpook National University
Daegu, Korea
sjkoh@knu.ac.kr

***Abstract*—The QUIC protocol offers a variety of distinctive features over TCP, which includes rapid handshaking process, zero-RTT capabilities, and connection migration. Among these, the connection migration plays a pivotal role in facilitating mobility for a myriad of mobile devices. However, the current implementation of connection migration necessitates the window size initialization when the IP address changes, such as during handovers. This leads to a decline in throughput performance. In this paper, we introduce an adaptive control of congestion window size when the client undergoes the connection migration due to IP changes. From the experimentation results, we can see that the proposed adaptive congestion control scheme provides significant performance improvement.**

Keywords—QUIC, Connection Migration, Congestion Window

I. Introduction

The QUIC (formerly known as Quick UDP Internet Connection) protocol has been developed by the IETF [1] as a cutting-edge transport protocol. The QUIC protocol encompasses numerous features, including a swift handshake integrated with TLS, Zero-RTT for transmitting user data during the handshake [2], and the ability for connection migration.

Nowadays, mobile devices, such as smartphones, tablets, and laptops, are widely used in the industry. As these devices are mobile, the operations of maintaining or reestablishing the connection's state during the connection is very crucial, due to its significant impact on user experience [3]. Therefore there are many reach to handle fast handover [4-7] In such scenarios, the connection migration can be especially valuable. Connection migration allows clients to sustain the connection's context regarding user data even when they relocate [1]. Consequently, if a client switches locations, its IP address might change. In this situation, while TCP would terminate the connection, QUIC maintains it through connection migration. This leads to a distinction in how the two protocols identify connections. TCP relies on IP addresses and port numbers from both the server and client pair for identification. However, QUIC uses a connection ID. Thus, if the IP address or port number changes as the client relocates, QUIC can still identify the connection.

However, an issue arises when connection migration occurs due to an IP address change. The current QUIC implementation encounters decreased throughput because it needs to reinitialize the current window size, which determines how many packets can be sent at once. While a change in port number maintains the window size, often observed in NAT rebinding, an IP address change might relocate the mobile device to a new network state. The client might not be aware if the new network state is stable or favorable.

To tackle this challenge, we propose an adaptive congestion control scheme for QUIC with the connection migration, in which the initial congestion window size (*cwnd*) is adaptively configured by considering the network conditions of the newly visited network, rather than by setting to the default value for the initial *cwnd*. Our proposed scheme retains a portion of the *cwnd* size instead of initializing it, determining the size based on a rough estimate of the bandwidth state in the new network. If the new network state significantly improves upon the previous one, the proposed scheme can mostly retain the window size. Conversely, if the state deteriorates, the proposed scheme initializes the window size due to the potential inability to handle the sending rate in the new network.

This paper is organized as follows. Section 2 present the proposed adaptive congestion control scheme for QUIC. In Section 3, we outline our experimental testbed for performance evaluation and compare the current QUIC scheme with our proposed scheme. Section 4 concludes this paper by suggesting avenues for future research.

II. Proposed Adaptive Control of Congestion Window

The proposed scheme introduces a method for addressing the initial congestion window when the client changes its IP address. Figure 1 illustrates the flow of the proposed scheme in this particular scenario.

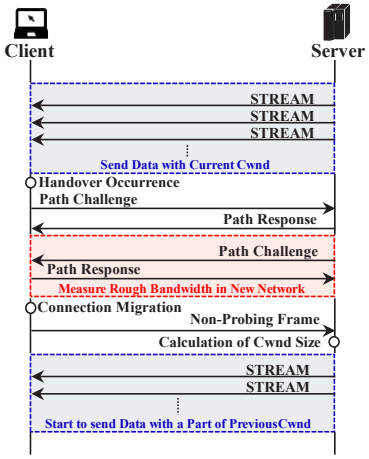

Fig. 1. Process of cwnd migration

A server transmits requested data to a client through STREAM frames in QUIC, based on the current congestion window. This window represents an approximate bandwidth estimation in the current network.

While the client receives data from the server, it encounters a handover event that potentially alters its IP address. Subsequently, the client initiates path validation by sending a path challenge frame in QUIC. This process, known as path validation within QUIC, has twofold: to confirm that the endpoint can reach the peer via the designated path, and to verify the peer remains that same despite any changes in the path. In this context, the client employs path validation to ensure its ability to reach the server. Upon the server receiving the path challenge, it promptly responds by transmitting the path response frame within the QUIC protocol. The path response contains an identical value, essentially functioning as a nonce, which mirrors the value found in the path challenge generated by the sender of the path challenge itself. Upon successful reception of the path response by the client, the path validation is deemed successful, signifying that the client can now transmit data to the server using this verified path.

However, it is important to note that path validation does not extend to validating a peer's ability to send data in the opposite direction. This limitation arises due to insufficient entropy and the potential for spoofing. Consequently, the server also initiates a path challenge. In this context, the server seeks to assess the round-trip time along this new path, which may involve utilizing the default path within the connection with the client.

Once the server receives a path response matching the path challenge it issued, the server gains the latest round-trip time in the new network. Additionally, the client proceeds with the migration of its connection to the new network, having successfully verified the path. Following the migration of the connection, the client dispatches Non-probing Frames, encompassing frame types within QUIC that are not probing in nature, such as ACK and STREAM frames. It's important to acknowledge that the server cannot initiate the transmission of non-probing frames until it receives corresponding non-probing frames from the client. Consequently, after the server receives these non-probing frames, it can commence transmitting data using STREAM frames.

Prior to transmitting frames, the server calculates the initial congestion window size using the latest round-trip time in the new network, as illustrated by the following equation:

$$\mathrm{cwnd}_{\mathrm{new}} = \max\left(\frac{RTT_{old} - RTT_{new}}{RTT_{old}}\,\mathrm{cwnd}_{\mathrm{old}},\ \mathrm{cwnd}_{\mathrm{init}}\right)$$

If the current round-trip time in the new network surpasses the minimum round trip time observed in the previous network, it suggests the potential for the network's performance to be less optimal in the new environment compared to the previous one, despite outward appearances. As a result, the initial window size is configured with a predetermined value chosen by the QUIC implementation.

However, if the latest round trip time is shorter than the minimal round trip time from the previous network, the server configures the initial congestion window based on the disparity between these round trip times. In essence, when the difference is considerable, the server might adopt an almost identical congestion window size. Conversely, if the difference is relatively small, the server may opt for the initial value of the congestion window.

After determining the initial congestion window size for the new network, the server initiates data transmission at a rate consistent with this window size.

III. Experimentation Results

A. Testbed setup for experimentation

Figure 2 depicts the testbed established to assess the proposed scheme. A router was configured to establish distinct paths to the server, which is connected to the KNU University Network. Additionally, we configured the smartphone to access the SKT 5G network through Ethernet tethering.

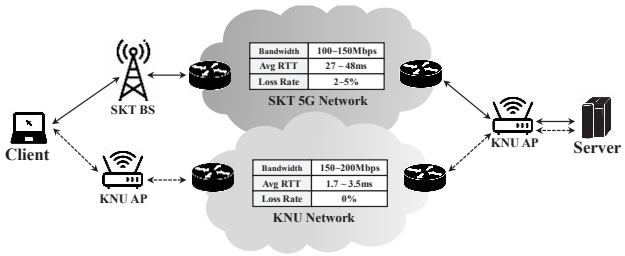

Fig. 2. Testbed configuration for experimentation

The KNU network environment encompasses a network bandwidth ranging from 150 Mbps to 200 Mbps, exhibiting a 0% packet loss rate, and an average round-trip time (RTT) spanning from 1.7ms to 3.5ms. The routers employed included the ASUS RT-AX55 and the IpTIME A8004T models. Furthermore, the SKT 5G network environment features a network with speeds ranging from 100 Mbps to 150 Mbps, with a packet loss rate varying between 0% and 5%. The average round-trip time within this network environment ranges from 27ms to 48ms.

We employed the Chromium QUIC transport module [8] and application on Ubuntu 20.04 with Linux kernel version 5.15.0-52. The STREAM frame typically consists of 1,250 bytes. Throughout data transmission, network handover events occur. To manage congestion, we employed the CUBIC algorithm [9], the default congestion control algorithm in Chromium.

B. Comparison of Congestion Window Size

Figure 3 depicts the congestion window size over the course of the experiment, representing a portion of the total transmission. During the experiment, "cwnd-initial" refers to the initial value of the congestion window, which is set to 32 as determined by Chromium when the network path undergoes a change. Conversely, "cwnd-migration" pertains to the proposed scheme, involving the migration of a portion of the previous network's congestion window size.

A handover event takes place at the 200 ms mark within the experiment timeline. Furthermore, in this experiment, we performed a handover from the SKT 5G network to the KNU network. Additionally, each value presented in the experiment timeline is an average derived from a total of 10 experimental runs.

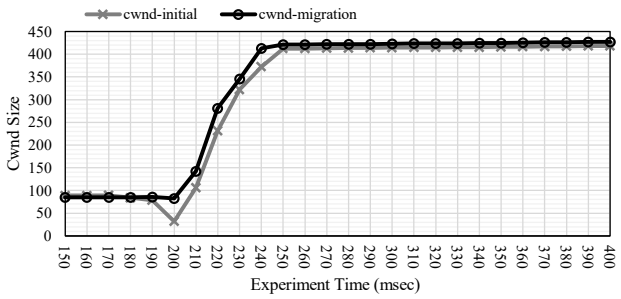

Fig. 3. Trace of congestion windows in two schemes

As the handover occurred, our proposed scheme maintained a relatively stable initial window size, slightly lower than the size prior to the handover event. Notably, the congestion window size for the new path reached around 86% of the size for the old path in this context. After the handover, our observations underscored the proposed scheme's agility in adjusting the congestion window size to match current bandwidth conditions, surpassing the performance of the existing scheme. This was evident in the larger window size of the proposed scheme between 200ms and 250ms compared to the existing scheme. As a result, we observed the proposed scheme achieving a stable window size faster than the existing one.

C. Comparison of Handover Thoughput

To compare the performance between the two schemes, we measured the handover throughput from the start of the handover until 100ms afterward. Figure 4 shows the handover throughputs for the existing *cwnd-init* scheme and the proposed *cwnd-migration* scheme.

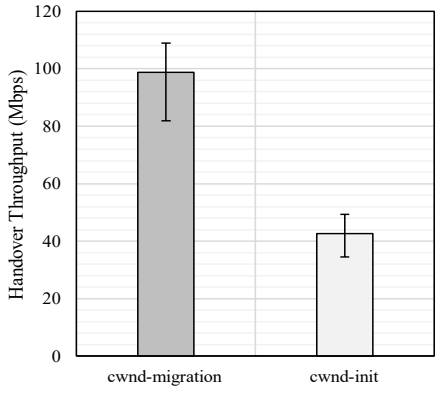

Fig. 4. Trace of congestion windows in two schemes

We observed a nearly doubled throughput in the proposed scheme, attributed to its capability to initiate data transfer at a rate closely aligned with the previous network's rate. We hypothesize that if the bandwidth difference were even greater than what our current testbed indicates, the proposed scheme might yield even higher throughput compared to the existing scheme.

IV. Conclusions with Future Works

This paper outlines our approach to maintaining consistent window sizes in QUIC despite IP changes. We address scenarios like IP changes during server data reception due to handovers. We introduce an equation to maintain window sizes from the previous network configuration. Additionally, we detail our performance evaluation setup, comparing our proposed scheme to an existing one. Our results reveal that the proposed scheme outperforms the existing approach. However, relying solely on single path validation for bandwidth assessment might be inadequate due to the limitation that a single round-trip time may not accurately depict the new network's bandwidth.

Thus, our upcoming research will focus on enhancing bandwidth measurement methods by employing specific measurement intervals after an IP change for more accurate results. We will also explore path selection policies, especially for clients with diverse paths through various network interfaces. Additionally, we'll compare this with MPQUIC, a protocol that utilizes multiple paths for data transfer [10].

Notably, we plan to conduct tests for our future research in an ultra-low latency SDN (Software-Defined Networking) network, allowing us to evaluate streaming service performance under ideal network conditions.

Acknowledgment

This work was supported by the Technology Innovation Program (20009633) funded by the Ministry of Trade, Industry & Energy (MOTIE, Korea) and Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (NRF2021R1I1A3057509).

References

[1] IETF Request for Comments 9000, QUIC: A UDP-based Multiplexed and Secure Transport, May 2021.

[2] IETF Request for Comments 9001, Using TLS to Secure QUIC, May 2021.

[3] A. C. -F. Chan, et al., "Impacts of handoff on TCP performance in mobile wireless computing," IEEE International Conference on Personal Wireless Communications, Mumbai, India, 1997, pp. 184-188.

[4] V. Tiwari, S. Kansal and A. Gaiwak, "Performance evaluation of TCP variants using Media Independent Handover in heterogeneous network," 2010 International Conference on Computer and Communication Technology (ICCCT), Allahabad, India, 2010, pp. 367-370.

[5] WonSeck Jung, SangWoo Son and ByungHo Rhee, "A study of enhanced TCP for vertical Handover using explicit congestion notification (ECN)," 2011 Third International Conference on Ubiquitous and Future Networks (ICUFN), Dalian, China, 2011, pp. 305-308.

[6] S. Pack, et al., "Fast-handoff support in IEEE 802.11 wireless networks," IEEE Communications Surveys & Tutorials, Vol. 9, No. 1, pp. 2-12, First Quarter 2007.

[7] Y. -S. Chen, et al., "DeuceScan: Deuce-Based Fast Handoff Scheme in IEEE 802.11 Wireless Networks," IEEE Transactions on Vehicular Technology, Vol. 57, No. 2, pp. 1126-1141, March 2008.

[8] The Chromium Project, QUIC: a multiplexed stream transport over UDP, 2023, Available: www.chromium.org/quic.

[9] IETF Request for Comments 8312, CUBIC for Fast Long-Distance Networks, February 2018.

[10] IETF QUIC Working Group, Multipath Extension for QUIC, draft-ietf-quic-multipath-03, October 2022