

Rescuing QUIC Flows From Countermeasures Against UDP Flooding Attacks

Junseok Lee
Dept. of Computer Science and
Engineering
Kangwon National University
Chuncheon, South Korea
jslee435@gmail.com

Minhyeong Kim
Dept. of Computer Science and
Engineering
Kangwon National University
Chuncheon, South Korea
1114pica@naver.com

Wonjun Song
Dept. of Computer Science and
Engineering
Kangwon National University
Chuncheon, South Korea
wjsong@kangwon.ac.kr

Younghoon Kim*
Dept. of Computer Science and
Engineering
Sungkyunkwan University
Suwon, South Korea
kyhoon@gmail.com

Dohyung Kim*
Dept. of Computer Science and
Engineering
Kangwon National University
Chuncheon, South Korea
mr.dhkim@gmail.com

ABSTRACT

Due to advantages such as quick connection establishment and multiple streaming over a single connection, QUIC was included in the new standard of HTTP 3.0 as an alternative transport layer protocol. Since QUIC operates on UDP, however, QUIC flows can be blocked by existing countermeasures against UDP flooding attacks, even if transmission rates are fairly controlled by congestion control algorithms, such as TCP. In this paper, we confirm that such a problem arises in real-world Internet environment and design effective approaches to avoid it. In the first approach, the gateway router dynamically sets the rate limit for the QUIC flow, based on the expected next CWND size estimated by the receiver using a built-in congestion control algorithm. The second approach leverages the proactive dropping of packets (or ECN marking) to distinguish whether the flow is a self-regulated QUIC flow or an unresponsive UDP attack/selfish flow. Simulation studies using the ns-3 simulator confirm that the proposed approaches can selectively allow QUIC flows regardless of their short-term transmission rates while preserving the effectiveness of existing countermeasures against UDP flooding attacks.

CCS CONCEPTS

• **Networks** → *Network management*; **Wired access networks**; **Network security**;

*corresponding authors

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

SAC '24, April 8–12, 2024, Avila, Spain

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-0243-3/24/04...\$15.00

<https://doi.org/10.1145/3605098.3635885>

KEYWORDS

UDP flooding attacks, Self-regulated QUIC flows, Network security, receiver-side RTT estimation in QUIC

ACM Reference Format:

Junseok Lee, Minhyeong Kim, Wonjun Song, Younghoon Kim, and Dohyung Kim. 2024. Rescuing QUIC Flows From Countermeasures Against UDP Flooding Attacks. In *The 39th ACM/SIGAPP Symposium on Applied Computing (SAC '24)*, April 8–12, 2024, Avila, Spain. ACM, New York, NY, USA, 9 pages. <https://doi.org/10.1145/3605098.3635885>

1 INTRODUCTION

Traditionally, web applications have been serviced using the HTTP protocol over TCP connections. An encryption protocol called Transport Layer Security (TLS) is added to provide privacy and data integrity. A three-way handshake of TCP [1] and the key exchange/agreement process of TLS must be performed to establish an HTTP connection, which has a significant effect on page load time (PLT). Besides, in-order delivery upon TCP may cause the head-of-line blocking (HOLB) problem in the HTTP pipelining, aggravating PLT.

In order to resolve the problems, Google has developed an alternative transport layer protocol called QUIC (Quick UDP Internet Connections) [2], which uses UDP as the transport layer protocol and support network functions for reliable and efficient content delivery at the application layer. Since QUIC operates at the top of UDP, the connection setup latency could be reduced to 0 or 1 round-trip time (RTT). On UDP, packet loss in a single stream does not block packet transmission across multiple other streams, thus avoiding HOLB. Consequently, QUIC reduces Google search latency by 8% on average and video re-buffer time by 18% for desktop [3]. To be TCP-friendly, QUIC offers a plug-type congestion control, which facilitates the application of various congestion control schemes.

Due to the characteristics of simplicity and low overhead, UDP packets are easily exploited to implement denial of service (DoS) attacks. Since a high volume of “best effort” traffic can be sent to any host without checking for valid connections, the receiving host may be overloaded and the network where the receiving host is

located can be severely congested. To protect receiving host and its network, firewalls are typically installed on gateway routers to filter out malicious UDP packets. Here it is noted that legitimate QUIC flows delivered using UDP packets can also be blocked due to high short-term transmission rates, even if they are fairly controlled by plug-type congestion control methods.

In this paper, we report that such a problem is occurring in real-world Internet environment and design two simple but effective solutions to address the problem. The first solution works in a collaborative manner between the gateway router (running a firewall) and the QUIC receiver. The QUIC receiver uses a built-in congestion control algorithm to estimate the sender's next congestion window size (transmission rate) and informs the gateway router of the expected maximum receivable sequence number. This allows the gateway router to dynamically set the appropriate rate limit, and as a result, QUIC packets may not be filtered out regardless of the short-term transmission rate. In the second solution, without any help of other nodes, the gateway router exploits the idea of Active Queue Management (i.e., the proactive dropping of packets) to determine whether the flow is self-regulated or not. When the incoming rate of a flow exceeds the rate limit, the gateway router intentionally drops a single packet of the flow and monitors the flow's incoming rate during a couple of round-trip time periods. If the incoming rate decreases, the gateway router determines the flow as a self-regulated one and prevents the rate limit from being applied to that flow.

The proposed solutions are implemented in the ns-3 simulator and their efficacy is evaluated. The simulation studies show that the proposed solutions allow QUIC flows to be selectively serviced regardless of their short-term transmission rates while preserving the effectiveness of the existing countermeasures against UDP flooding attacks. The rest of this paper is organized as follows. The related works and the motivation for our work are addressed in Section 2. In Section 3, the design of the proposed scheme is described, and the effectiveness of these schemes is verified in Section 4. Section 5 concludes this paper.

2 RELATED WORK AND MOTIVATION

2.1 QUIC

It has been about 30 years since HyperText Transport Protocol (HTTP), the underlying protocol for World Wide Web (WWW), was first designed. Using HTTP, web resources such as servers' HTML files have been accessed, and they appeared on the web browsers with text and images. Over the years, the Internet has evolved considerably, and many websites are offering much more resource-intensive content than ever. Even native applications are being delivered in the web environment.

Amid these changes, the initial versions of HTTP, known as HTTP 0.9 ~ 1.1, reveals their limitations. Up to HTTP 1.0, a new TCP connection is created for each request, which increases the latency and overhead of opening connections to multiple queries. From the version of HTTP 1.1, instead, a persistent connection is established between the server and the client, and it is reused for successive requests. In order to further reduce the PLT, a number of successive requests are sent without waiting for responses, which is called HTTP pipeline technology. Here, not all HTTP requests

could be pipelined and requests that should be processed for a long time may hold up subsequent responses, resulting in a HOLB.

To address all of these limitations and speed up web services, HTTP/2.0 was introduced in 2015 [4]. Its design is based on SPDY, a previous HTTP replacement protocol developed by Google [5]. HTTP/2.0 supports multiplexing of multiple data streams over a single TCP connection. Therefore, it resolves the problem of HOLB at the HTTP level. However, since multiple data streams are sent over a single TCP connection, HTTP/2.0 cannot resolve the TCP HOLB, where loss or delay of some TCP segments on the TCP head of line makes subsequent segments be held by the TCP stack.

In 2013, Google introduced QUIC, an experimental transport protocol for the purpose of realizing a faster web [6]. QUIC provides the functionality of TCP + TLS + HTTP but is layered on top of UDP instead of TCP (Fig. 1). Therefore, services may be no longer delayed due to TCP routines such as three-way handshake and in-order delivery. The distinguishing features of QUIC that can affect PLT are summarized as follows [7].

- (1) Fast connection establishment : Since in TCP-based web services, data is transferred over the connection established by performing TCP three-way handshake and TLS handshake, a delay of three RTTs is inevitable before data transmission. In HTTP 3.0 using QUIC, application data can be transmitted after at most one RTT required for exchanging security information (creating the QUIC connection.).
- (2) Multiplexing without Head-of-Line blocking : In order to avoid HOLB due to TCP's in-order delivery [8], QUIC supports multiplexing multiple streams over a UDP connection. Each data is carried in a stream frame that is identified by a unique 64-bit stream ID, and multiple frames carrying data independent of one another (corresponding to multiple streams) are encapsulated in a single QUIC packet. The stream is implicitly created by using a frame with a new (unused) stream ID and is terminated by setting a "FIN" on the last frame of the stream when data forwarding is complete and the stream is no longer needed. Each stream is treated as a logically independent flow so that clients can continue to process data from other streams even if packets in a certain stream are dropped.
- (3) plug-type congestion control : Since QUIC packets are sent over UDP connections, the reliability mechanism provided by TCP (e.g., TCP congestion/flow control, in-order delivery, etc.) is not available. To address the issue, QUIC implements its own reliability mechanism in the user space, not in the kernel space. In particular, congestion control is provided as a plugin to make QUIC flows TCP-friendly. Algorithms for TCP performance enhancement (e.g., FACK [9], TLP [10], F-RTO [11], Early retransmit [12], PRR [13], etc) are also incorporated with QUIC.

2.2 UDP flooding attacks

The UDP flooding attack is a type of DoS (Denial of Service) attack where a massive amount of UDP packets are sent to a target host or network. Unlike TCP, which self-regulates the offered load, UDP has no congestion control mechanism. This allows large amounts of data to flow into the network using UDP flows, which can be exploited to launch attacks that waste network bandwidth and computing resources.

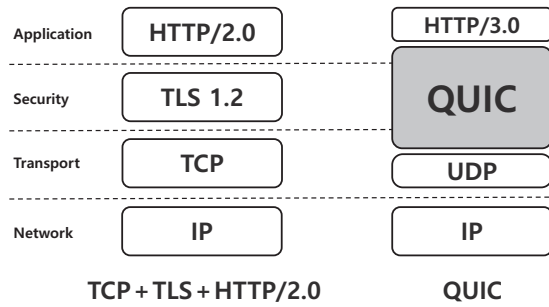


Figure 1: HTTP/2 vs. HTTP/3

In order to handle UDP flooding attacks, in general, network traffic is monitored in real-time on the gateway router and suspicious flows are selectively blocked. Here, the detection of attack flows has become an important research issue, and a variety of schemes have been presented. In particular, the authors of [14–17] presented schemes for identifying attack flows using the proportional packet rate¹. More specifically, the traffic rates in both directions are generally known to be proportional [14]. Under flooding attacks, the rate of traffic entering the victim network becomes much higher than that of traffic leaving the victim network, resulting in larger traffic asymmetry. Hence, larger traffic asymmetry may indicate that a flooding attack is being launched. However, the authors of [18] pointed out that even in multimedia streaming services, traffic asymmetry can be large since the traffic rate from the server is much higher than that from the client, and thus the method of using proportional packet rate can generate a high false positive rate. Moreover, since the proportional packet rate in an attack flow can be manipulated similarly to the proportional packet rate in a legitimate flow, traffic asymmetry may not be an effective metric for detecting attack flows.

2.3 Motivation

The QUIC flow may have a legitimate high short-term transmission rate similar to TCP flows, depending on the congestion control algorithm. However, since QUIC operates on UDP, the QUIC flow could be considered a UDP flooding attack on the gateway router and may be blocked by countermeasures against UDP flooding attacks. Fig. 2 shows the rate of data reception captured by Wireshark when a user on a campus network requests an 8K Ultra HD/60FPS YouTube video. The user starts receiving data at 5.3th second, but suddenly stops receiving data around 6.45th second. The firewall installed on the gateway router counts the number of UDP packets (per second) directed to each host and blocks a UDP flow if the number of packets belonging to the flow exceeds a certain threshold. Due to this operation of the firewall, QUIC packets are discarded from around 6.45th second.

Our observation was reaffirmed by the network administrators. Their report says UDP flows with a transmission frequency greater than 512 during 4 seconds could be blocked, and recently this threshold value has been manually increased to 2048 transmissions / 4

¹The ratio of incoming and outgoing traffic rate

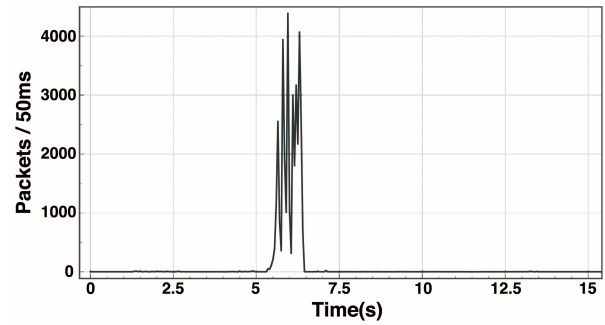


Figure 2: Receiving 8K Ultra HD/60FPS YouTube video at SKKU campus

seconds. Here it is noted that no matter what the threshold is set to, the congestion control scheme (in both TCP and QUIC) can increase the transmission rate to exceed that threshold, depending on the available bandwidth. Hence, a manual configuration with high thresholds cannot solve the problem properly.

Several methods can be suggested as an alternative to this problem, but their limitations can be discussed as follows.

- Currently, UDP port 443 is being used to deliver QUIC packets. It may be a stopgap to ensure that flows with UDP port 443 are not regulated by the firewall's rate limit. However, packets created by flooding attacks may also have the same port number.
- By registering IP addresses of well-known content providers, packets sent from these providers can be exceptionally prevented from being blocked. With the inclusion of QUIC in the HTTP 3.0 standard, however, more and more naive users would use QUIC as an alternative transport layer protocol, and thus this approach may be temporary and incomplete.
- Typically, UDP flooding attacks generate unidirectional massive traffic, while QUIC flows have both incoming and outgoing traffic due to the receiver feedback-based reliability mechanisms. Based on this observation, the firewall might not enforce rate limits on flows with bidirectional traffic. However, the authors of [19] reported that this approach could be vulnerable to the “UDP Hole Punching” attack, in which packets from the compromised node (in the internal network) are used to “punch a hole” in the firewall.

3 THE PROPOSED SCHEMES

To ensure that QUIC flows are not harmed by the countermeasures against UDP flooding attacks, this paper proposes two approaches – the collaborative approach and the AQM²-based approach. In the collaborative approach, the QUIC receiver uses a built-in congestion control algorithm to estimate the expected traffic volume in the next RTTs. Based on this, the allowable amount of traffic is calculated and proactively notified to the gateway router with the firewall installed, which enables dynamic configuration of per-flow traffic

²Active Queue Management

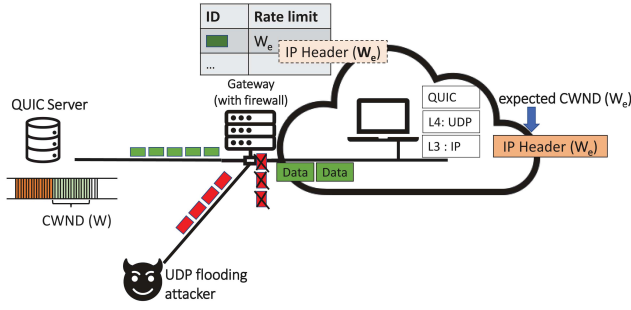


Figure 3: Collaborative approach

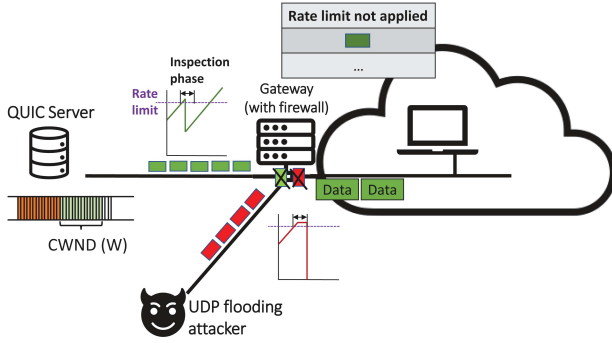


Figure 4: AQM-based approach at the gateway router

limits (Fig. 3). In the AQM-based approach, the gateway router intentionally drops a single packet (or marks ECN³) of the UDP flow whose short-term transmission rate exceeds the threshold, and determines the flow as a self-regulated flow if the incoming rate decreases during the next a couple of RTT periods. If the UDP flow is identified as a self-regulated one, it would not be affected by countermeasures against UDP flooding attacks (Fig. 4).

3.1 The collaborative approach

To implement the idea, the following questions should be answered - 1) How is the traffic volume of the incoming QUIC flow estimated in advance? 2) How does the gateway router know the estimated traffic volume?

3.1.1 How the legitimate incoming rate is estimated. As mentioned above, QUIC uses receiver feedback-based reliability mechanisms and window-based congestion control mechanisms, such as TCP. Accordingly, on the receiver side, a change in the congestion window of the sender may be predicted in advance based on the built-in congestion control algorithm and the acknowledgement packets to be generated. Let W_e and W be the congestion window size

³Explicit Congestion Notification

Algorithm 1 The collaborative approach at receiver

```

1:  $\mathcal{F} \leftarrow$  built-in function to compute expected traffic (CWND)
2:  $\mathcal{T} \leftarrow$  total traffic amount that has been received for a flow
3:
4: if QUIC packet is received then
5:    $\text{fid} \leftarrow$  a flow ID extracted from  $\mathcal{P}$ 
6:    $\text{fid's } \mathcal{T} \leftarrow \text{fid's } \mathcal{T} + \text{bytes of } \mathcal{P}$ 
7:   if ACK packet is generated then
8:      $W_e \leftarrow \mathcal{F}(\text{ACK})$ 
9:     send ACK with  $\leftarrow \mathcal{T} + W_e$ 
10:  end if
11: end if

```

Algorithm 2 The collaborative approach at firewall

```

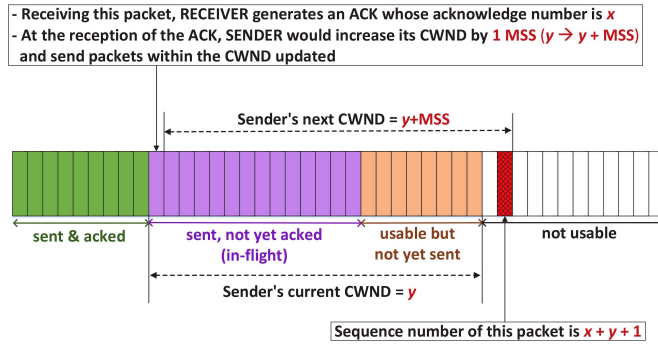
1:  $\mathcal{T} \leftarrow$  total traffic amount that has been received for a flow
2:  $\mathcal{L} \leftarrow$  allowable Traffic amount for a flow (In general,  $\mathcal{L} > \mathcal{T}$ )
3:
4: if Incoming packet with 443 port ( $\mathcal{P}$ ) is received then
5:   if  $\mathcal{P}$  is a QUIC data packet then
6:     run ApplyVolumeLimit ( $\mathcal{P}$ )
7:   end if
8: else if Outgoing packet with 443 port ( $\mathcal{P}'$ ) is received then
9:   if  $\mathcal{P}'$  is a QUIC ACK packet with  $(W_e + \mathcal{T})$  then
10:    run UpdateVolumeLimit ( $\mathcal{P}'$ )
11:   end if
12: end if
13:
14: procedure APPLYVOLUMELIMIT( $\mathcal{P}$ )
15:    $\text{fid} \leftarrow$  a flow ID extracted from  $\mathcal{P}$ 
16:    $\text{fid's } \mathcal{T} \leftarrow \text{fid's } \mathcal{T} + \text{bytes of } \mathcal{P}$ 
17:   if  $\text{fid's } \mathcal{T} > \text{fid's } \mathcal{L}$  then
18:     block the flow of  $\text{fid}$ 
19:   end if
20: end procedure
21:
22: procedure UPDATEVOLUMELIMIT( $\mathcal{P}'$ )
23:    $\text{fid} \leftarrow$  a flow ID extracted from  $\mathcal{P}'$ 
24:   if  $\text{fid}$  is not found in the table then
25:     create an entry for  $\text{fid}$ 
26:   end if
27:    $\text{fid's } \mathcal{L} \leftarrow (W_e + \mathcal{T})$  extracted from  $\mathcal{P}'$ 
28: end procedure

```

(CWND) expected on the receiver and the actual CWND of the sender, respectively. If the gateway router pre-recognizes W_e for a QUIC flow, it can set an appropriate limit to accommodate all legitimate packets while preventing flooding attacks. The detailed operations are described in Algorithm 1.

The following considerations should be taken into account when calculating W_e .

- Loss of ACK packets: Let $W_e(t)$ and $W(t)$ be the expected CWND (estimated by the receiver) and the actual CWND (set by the sender) for t -th RTT round. Since acknowledgement packets could be lost during delivery, $W(t)$ may be temporarily smaller than $W_e(t)$. In this case, however, the amount of traffic that



* RTT estimation at the receiver : Time between sending an ACK with an ack. no. of x and receiving a packet with a seq. no. of $x+y+1$

Figure 5: RTT Estimation at the QUIC receiver

the sender would send is smaller than the limit set by the gateway router ($W(t) < W_e(t)$), and all legitimate packets can pass through the gateway router.

- RTT : In QUIC, when a packet is lost or congestion is explicitly notified, the sender goes into the state of “recovery period” which lowers the sending rate by reducing the CWND. It is noted that even if a new packet loss or a new congestion notification occurs during the recovery period, the CWND is not further reduced. That is, the reduction of CWND is allowed only once per RTT [20]. In order to estimate the sender’s CWND accurately, therefore, the receiver should be aware of each flow’s RTT. In the proposed scheme, the receiver estimates RTT in the following way. When an ACK generated at the receiver would increase the CWND by one maximum segment size (MSS), the receiver records the acknowledgement number of the ACK (x) and the increased CWND ($y + \text{MSS}$). If the ACK is delivered to the sender, the sender would transmit packets whose sequence number is smaller than or equal to $x+y+1$ in response. Hence, at the receiver, the RTT can be estimated from the time when sending an ACK (with the acknowledgement number of x) that would increase the CWND by one MSS (resulting in the CWND size of $y+\text{MSS}$) to the time when receiving the packet whose sequence number is $x+y+1$ (Refer to Fig. 5). During the estimated RTT, multiple CWND reductions are not allowed at the receiver as well.

3.1.2 How the gateway recognizes the legitimate incoming rate. The remaining work is to inform the gateway router of the amount of allowable traffic estimated by the receiver. For QUIC flows, fortunately, the receiver sends an acknowledgement packet to the sender as feedback. Hence, the estimated value is included in the option field of the IP header (IPv4) or in the extension header (IPv6), which allows the gateway router to set the appropriate volume limit for the QUIC flow. The detailed operations are described in Algorithm 2.

If, for each packet, the gateway router is notified of the sender’s new CWND and updates the incoming rate of the flow, the overhead may be significant. In order to reduce the overhead, this process

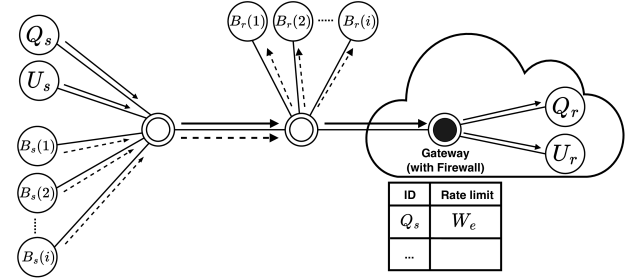


Figure 6: Simulation topology

Algorithm 3 AQM-based approach

```

1: if A UDP packet is received then
2:   if The flow exceeds the rate limit then
3:      $\text{fID} \leftarrow$  the flow ID
4:     search fID in the ExceptionList
5:     if the matching entry is not found then
6:       run InspectionPhase(fID)
7:     end if
8:   end if
9: end if
10:
11: procedure INSPECTIONPHASE(fID)
12:   discard a fID' packet (or set CE bit on a fID' packet)
13:   monitor fID's incoming rate for a period
14:   if fID's incoming rate < fID's rate limit then
15:     add the flow to the ExceptionList with expiration time
16:   else
17:     block the flow of fID
18:   end if
19: end procedure

```

can be performed every n RTT rounds. In that case, the receiver assumes that no packet loss occurs during the next n RTT rounds, and estimates the amount of traffic that can be received until after n RTT rounds. That is, the receiver estimates $W_e(t+n)$ at t -th RTT round and calculates the maximum amount of traffic to be allowed until the next n RTT rounds. This value is used at the gateway router as the volume limit during the next n RTT rounds. This strategy demonstrates the tradeoff between overhead efficiency and security. Reducing overhead can temporarily compromise security.

3.2 AQM-based approach at firewall

Generally, routers accommodate queues for buffering packets due to differences in packet arrival rates and packet departure rates. When the network is congested, the queue of the router may become full, and in this case, newly arrived packets may be discarded in a burst manner. The burst packet drop can significantly reduce the throughput of TCP flows. To address the problem, AQM has been introduced in [21–24]. In AQM, the router can pre-drop packets before the queue becomes full. This proactive dropping of packets allows TCP senders to reduce the transmission rate, which in turn

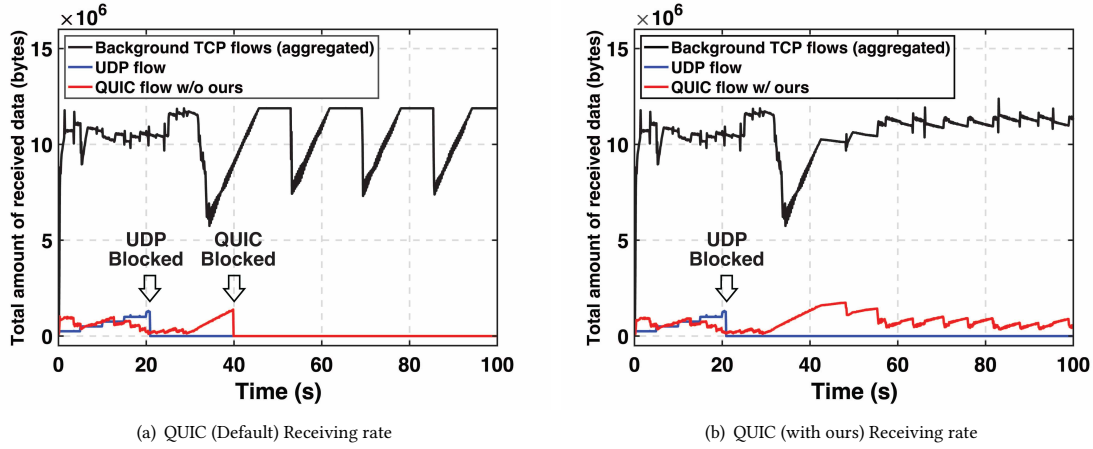


Figure 7: Collaborative approach against UDP and QUIC flows

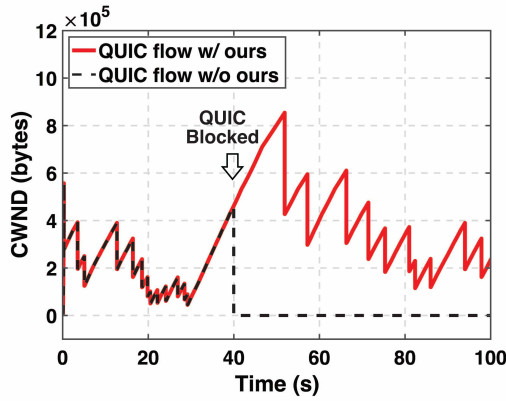


Figure 8: QUIC CWND

can alleviate network congestion. Instead of dropping a packet, routers can also set a mark for Congestion Encountered (CE)⁴ to inform the sender of network congestion.

Here we note that self-regulated flows slow down the transmission rate in response to packet loss (ECN mark). The proposed approach uses this fact to identify whether the flow is a self-regulated QUIC flow or an unresponsive attack flow. If the incoming rate of a UDP flow exceeds the rate limit of the firewall, the flow will go through the inspection phase. In the inspection phase, the router discards a single UDP packet (or sets CE bits) and then monitors the incoming rate of the flow for a couple of RTT periods. If the incoming rate decreases and becomes smaller than the rate limit, the firewall creates an exception to prevent the flow from being blocked by the rate limit. Then, subsequent packets belonging to the flow can be forwarded through the gateway router regardless of the short-term transmission rate. The inspection phase can be repeated after a random time interval to achieve better security,

⁴two least significant bits of the Traffic Class field in the IP header are set to 11

which poses a tradeoff relationship between the throughput of flows and security. The detailed operations of this approach are described in Algorithm 3.

Compared to the collaborative approach, this approach has the advantage of not needing help from other nodes and not generating message overhead. However, the throughput of QUIC flows could be slightly damaged due to the intentional packet drop (or ECN mark) during the inspection phase.

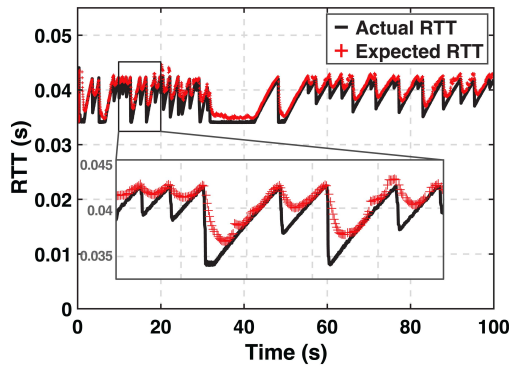
4 EVALUATION

The proposed scheme is implemented on the ns-3 simulator [25] and evaluated via simulation studies. Fig. 6 shows the simulation topology, where the node Q_r and U_r behind the gateway router receive QUIC and UDP packets from the node Q_s and U_s , respectively. The background TCP traffic is transmitted from $B_s(i)$ to $B_r(i)$, for $i = 1 : 20$. All links have a bandwidth of 1Gbps.

4.1 The effectiveness of the collaborative approach

Fig. 7 shows the amount of data received for UDP and QUIC flows under the collaborative approach. Initially, UDP packets are transmitted at a rate of 20Mbps, and the transmission rate increases by 20Mbps every 5 seconds. When the transmission rate reaches 100 Mbps, the firewall on the gateway router begins to discard UDP packets destined for the U_r . Fig. 7(a) shows that the firewall also drops packets of the QUIC flow for the same reason. With 13 background TCP flows disappearing at 30th second, Q_s 's CWND begins to increase and exceeds the threshold set in the firewall (100Mbps) at around 40th second (Fig. 8), resulting in QUIC flow being blocked. With the proposed approach, however, it is confirmed that the QUIC flow can survive from the firewall, and its transmission rate is affected/controlled only by the congestion control mechanism of the QUIC (refer to Fig. 7(b) and Fig. 8). Here, Fig. 9 shows the effectiveness of the proposed RTT estimation scheme. The sender measures an RTT value for each packet, while the receiver estimates the RTT

	Throughput (MB)	
	Collaborative approach	AQM-based approach
QUIC flow1	888.55	857.40
QUIC flow2	911.16	786.09
QUIC flow3	787.52	807.02
QUIC flow4	783.78	788.69
QUIC flow5	726.52	854.00
TCPs (aggregate)	6,696.84	6,601.89
Total	10,794.36	10,695.08

Table 1: Collaborative approach vs. AQM-based approach**Figure 9: RTT estimation at Sender and Receiver**

once per CWND. Because of the difference, a marginal difference is shown between the two estimated RTTs, but it can be seen that, in general, the two RTT values change similarly.

The overhead of this approach can be effectively mitigated by increasing the notification interval. Fig. 10 shows the results when the expected sequence number is notified to the gateway router at different frequencies (every one, hundred and thousand RTT rounds, respectively). As the frequency of notification decreases, the temporal gap between the actual sequence number (sent by the sender) and the expected sequence number (registered at the gateway) increases. However, this gap set to the marginal value does not allow serious security vulnerability, but trade-off relationship.

4.2 The effectiveness of the AQM-based approach

In this section, the AQM-based approach is validated with the same simulation scenario mentioned above. When the incoming rate of the UDP flow reaches the rate limit (set to 100Mbps), under the proposed scheme, the flow goes into the inspection phase where a single packet is intentionally discarded and the incoming rate is monitored. Since the gateway router observes that the flow's incoming rate does not decrease in response to the packet loss, the flow is determined as an attack flow and blocked (refer to Fig. 11(a)). In contrast, the QUIC flow reacts to the packet loss and decreases

its transmission rate, as shown in Fig. 11(a) and Fig. 11(b). The gateway router then registers information about the flow, allowing the flow's incoming rate to exceed the rate limit. Here it is noted that if an attacker detects the rate limit (100Mbps in this simulation) in advance, the attacker can take the initiative in reducing the sending rate to trick the firewall. To avoid the problem, the firewall repeats the inspection phase after a random time interval (An interval of 10 seconds is selected for clarity of the resulting graph, but in reality much smaller value may be chosen) regardless of the flow's incoming rate. Through these two inspection phases, the firewall verifies that the flow is a self-regulated QUIC flow, and does not block the flow even if the flow's incoming rate exceeds the rate limit.

The effectiveness of the two approaches is evaluated with multiple QUIC flows (i.e., 2 UDP flows, 5 Quic flows, and 7 TCP flows), and the results are shown in Fig. 12 and Table. 1. It is confirmed that both proposed schemes can preserve legitimate QUIC flows while effectively presenting UDP flooding attacks. The inspection phase of the AQM-based approach does not significantly degrade the achieved throughput of QUIC flows, and thus, the total throughput of QUIC flows during the simulation time is very similar in both schemes.

5 CONCLUSION

The QUIC, developed by Google, is widely used as an alternative transport layer protocol for web applications and has recently been included in the HTTP 3.0 standard. This paper demonstrates that countermeasures against UDP flooding attacks in real-world Internet environments can block QUIC flows operating on UDP. To address the problem, this paper proposes two simple but effective approaches - collaborative approach and AQM-based approach. In the collaborative approach, the receiver estimates the sender's CWND using a built-in congestion control algorithm and notifies it to the gateway router, which allows the gateway router to accept the QUIC's self-regulated traffic regardless of the amount of incoming traffic. In the AQM-based approach, the gateway router intentionally drops a single packet and determines whether the flow is self-regulated or not based on the throughput monitored during a couple of RTTs. Via the simulation study using the ns-3 simulator, it is confirmed that the proposed approaches may selectively service QUIC flows while preserving the effectiveness of countermeasures against UDP flooding attacks.

REFERENCES

- [1] J. Postel, et al., Transmission control protocol rfc 793 (1981).
- [2] R. Hamilton, J. Iyengar, I. Swett, A. Wilk, et al., Quic: A udp-based secure and reliable transport for http/2, IETF, draft-tsvwg-quic-protocol-02 (2016).
- [3] A. Langley, A. Riddoch, A. Wilk, A. Vicente, C. Krasic, D. Zhang, F. Yang, F. Kouranov, I. Swett, J. Iyengar, et al., The quic transport protocol: Design and internet-scale deployment, in: Proceedings of the conference of the ACM special interest group on data communication, 2017, pp. 183–196.
- [4] M. Belshe, R. Peon, M. Thomson, Hypertext transfer protocol version 2 (http/2), Tech. rep. (2015).
- [5] B. Thomas, R. Jurdak, I. Atkinson, Spdying up the web, Communications of the ACM 55 (12) (2012) 64–73.
- [6] T. C. Projects, Quic, a multiplexed transport over udp, <https://www.chromium.org/quic/>.

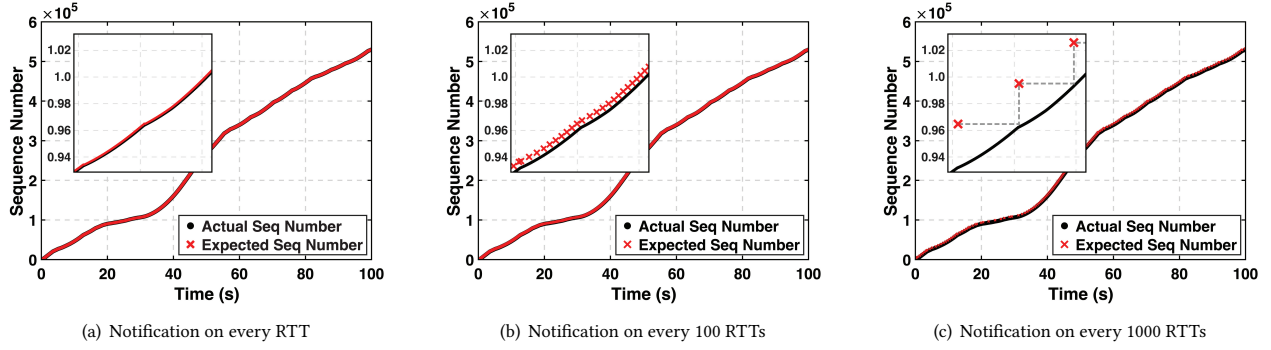


Figure 10: Different frequencies of notifications

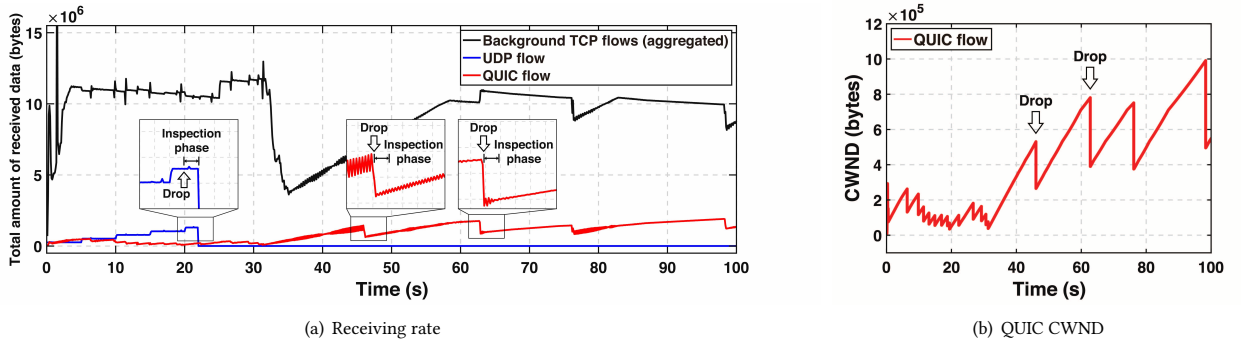


Figure 11: AQM-based approach against UDP and QUIC flows

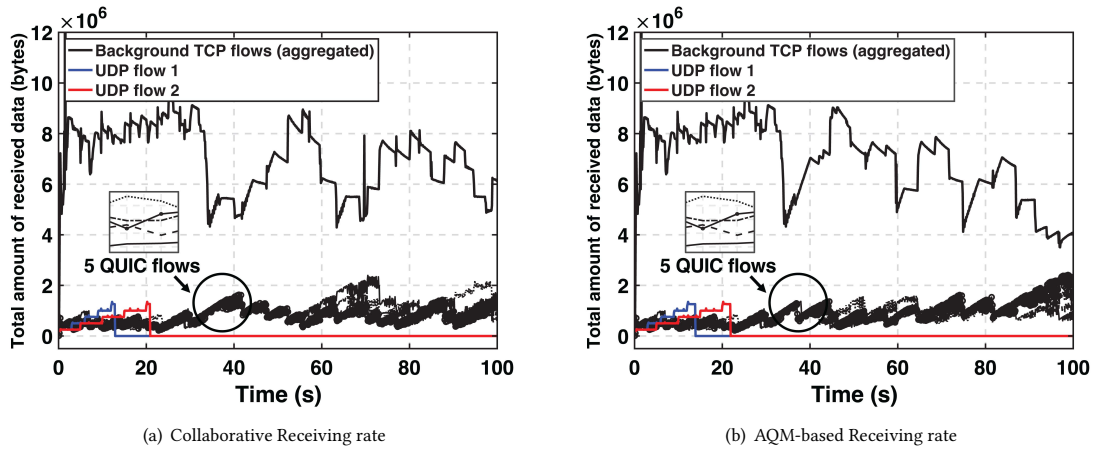


Figure 12: Evaluation with multiple QUIC flows

[7] J. Iyengar, M. Thomson, Rfc 9000 quic: A udp-based multiplexed and secure transport, Omtermet Emgoeomromg Task Force (2021).

[8] F. Qian, V. Gopalakrishnan, E. Halepovic, S. Sen, O. Spatscheck, Tm3: Flexible transport-layer multi-pipe multiplexing middlebox without head-of-line blocking, in: Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies, 2015, pp. 1–13.

- [9] M. Mathis, J. Mahdavi, Forward acknowledgement: Refining tcp congestion control, *ACM SIGCOMM Computer Communication Review* 26 (4) (1996) 281–291.
- [10] N. Dukkipati, N. Cardwell, Y. Cheng, M. Mathis, Tail Loss Probe (TLP): An Algorithm for Fast Recovery of Tail Losses, Internet-Draft draft-dukkipati-tcpm-tcp-loss-probe-01, Internet Engineering Task Force, work in Progress (Feb. 2013). URL <https://datatracker.ietf.org/doc/draft-dukkipati-tcpm-tcp-loss-probe/01/>
- [11] P. Sarolahti, M. Kojo, K. Raatikainen, F-rto: an enhanced recovery algorithm for tcp retransmission timeouts, *ACM SIGCOMM Computer Communication Review* 33 (2) (2003) 51–63.
- [12] M. Allman, K. Avrachenkov, U. Ayesta, J. Blanton, P. Hurtig, Early retransmit for tcp and stream control transmission protocol (sctp), Tech. rep. (2010).
- [13] N. Dukkipati, M. Mathis, Y. Cheng, M. Ghobadi, Proportional rate reduction for tcp, in: *Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference*, 2011, pp. 155–170.
- [14] T. M. Gil, M. Poletto, {MULTOPS}: A {Data-Structure} for bandwidth attack detection, in: *10th USENIX Security Symposium (USENIX Security 01)*, 2001.
- [15] S. Abdelsayed, D. Glimsholt, C. Leckie, S. Ryan, S. Shami, An efficient filter for denial-of-service bandwidth attacks, in: *GLOBECOM'03. IEEE Global Telecommunications Conference (IEEE Cat. No. 03CH37489)*, Vol. 3, IEEE, 2003, pp. 1353–1357.
- [16] A. G. Bardas, L. Zomlot, S. C. Sundaramurthy, X. Ou, S. R. Rajagopalan, M. R. Eisenbarth, Classification of udp traffic for ddos detection., in: *LEET*, 2012.
- [17] B. K. Devi, K. Dudeja, A. John, K. Marcin, T. Subbulakshmi, Symmetric measure of network traffic using packet ratio and packet symmetry, *Procedia Computer Science* 165 (2019) 112–118.
- [18] S. T. Zargar, J. Joshi, D. Tipper, A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks, *IEEE communications surveys & tutorials* 15 (4) (2013) 2046–2069.
- [19] K. Y. Gbur, F. Tschorsch, A quic (k) way through your firewall?, *arXiv preprint arXiv:2107.05939* (2021).
- [20] J. Iyengar, I. Swett, QUIC Loss Detection and Congestion Control, RFC 9002 (May 2021). doi : 10.17487/RFC9002. URL <https://www.rfc-editor.org/info/rfc9002>
- [21] S. Floyd, V. Jacobson, Random early detection gateways for congestion avoidance, *IEEE/ACM Transactions on networking* 1 (4) (1993) 397–413.
- [22] V. Firoiu, M. Borden, A study of active queue management for congestion control, in: *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No. 00CH37064)*, Vol. 3, IEEE, 2000, pp. 1435–1444.
- [23] S. Athuraliya, V. H. Li, S. H. Low, Q. Yin, Rem: Active queue management, in: *Teletraffic Science and Engineering*, Vol. 4, Elsevier, 2001, pp. 817–828.
- [24] W.-c. Feng, K. G. Shin, D. D. Kandlur, D. Saha, The blue active queue management algorithms, *IEEE/ACM transactions on networking* 10 (4) (2002) 513–528.
- [25] G. F. Riley, T. R. Henderson, *The ns-3 network simulator*, in: *Modeling and tools for network simulation*, Springer, 2010, pp. 15–34.

ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIP) (NRF-2020R1F1A1048395, 2020R1A4A3079947, 2021R1C1C1009778) and the Innovation Human Resource Development for Local Intellectualization support program (IITP-2023-RS-2023-00260267) supervised by the IITP (Institute for Information & communications Technology Planning & Education).