A New Low-rate DoS Attack Method Robust to Timing Skew for TCP Short Transfers

Ryuku Hisasue*, Hiroshi Inamura†, Shigemi Ishida†
* Graduate School of Systems Information Science, Future University Hakodate, Japan
Email: g2122054@fun.ac.jp
† School of Systems Information Science, Future University Hakodate, Japan
Email: {inamura, ish}@fun.ac.jp

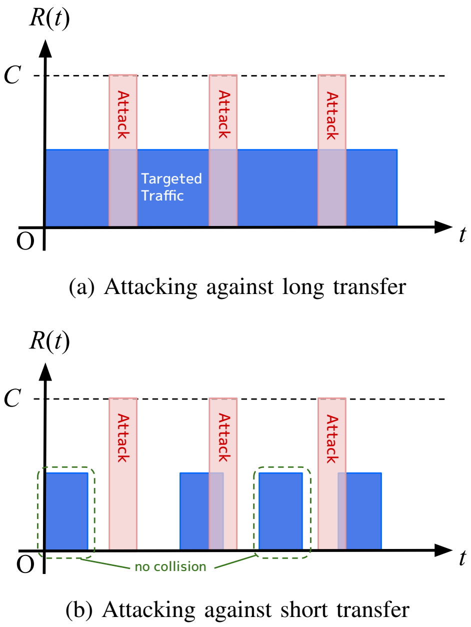
***Abstract*—It is posited that Low-rate DoS (LDoS) attack against TCP short transfer has not yet been studied. Since LDoS attacks use pulse-shaped attack traffic, the probability of an attack pulse colliding with the targeted traffic is low when the transfer time is short. In this study, we investigated the feasibility of the Shrew LDoS method against short transfers. In the Shrew method, the time difference between the targeted traffic and the attack traffic, i.e., the attack timing skew, has a large impact on the attack effectiveness for short transfers. Therefore, we proposed the First-Attack Pulse Width Expansion Shrew (Fawe-Shrew) method to improve the attack effectiveness in the presence of this skew. We confirmed that the proposed method has increased tolerance of the skew in the attack initiation timing.**

***Index Terms*—Low-rate DoS attack, Retransmission timer, Short transfer, Timing skew, Network security**

I. Introduction

Since 2003, Low-rate DoS (LDoS) attacks have been discussed as one of the major cyber-attacks throughout the world. LDoS attacks exploit algorithms used in protocols on computer networks by using periodic pulse traffic to achieve their attacks, so LDoS attack has stealthy evading classic network-based DDoS attack detection systems. It is the reason the victim cannot recognize the attack even if they are attacked [1].

To the best of the authors' knowledge, there have been no discussions on the LDoS attack for TCP short transfers in which the transfer is completed in a short period. In existing studies [2–9], the attack targets long transfers that occur when large amounts of data are transferred using FTP or other methods. LDoS attacks send burst traffic momentarily to the TCP link of the victim with the period which can exploit the vulnerability of protocol, and the TCP segment is lost by congestion [2–6, 10]. For happening congestion, attackers have to cause a *traffic collision* by the attack pulse and the attack traffic exists in the router queue at the same time. If the target sends large traffic, the attacker doesn't need estimated attack timing due to the long transfer time. However, as shown in Figure 1, making traffic collisions is difficult when the target is short transfers because attack start timing estimation is needed.



(a) Attacking against long transfer

(b) Attacking against short transfer

Fig. 1. Sending rate of targeted traffic and attack traffic. Timing estimation is not needed when the target is long transfer (a), but short transfer needs the estimation for making traffic collision (b).

If the timing of the attack pulses cannot be timed to coincide with the short transfer period, no traffic collision happens and the attack will fail because the target transfer will be finished before the attack pulses have been sent. Generally, it is difficult to intercept the communication contents and use the information to set the timing of the attack starting due to the encryption of the communication contents. However, it is more feasible to detect the execution of a 3-way handshake (HS) processing based on the TCP header information rather than intercepting the communication contents. For example, it is possible to detect HS processing when a connection is reestablished due to a session timeout. In fact, a cyber attack technique called active session hijacking exploits this characteristic to execute attacks [11]. This means that the HS process can be used as one of the methods to estimate the good timing for the success of the attack, and can be used as a trigger to start the attack. However, when using HS processing to estimate the attack start timing, the latency in the communication environment and the processing time until the actual transmission after HS processing may cause

doi: 10.1109/ICUFN57995.2023.10201096

a discrepancy from the actual attack start timing, as shown in Figure 1. The estimated attack timing error varies depending on the environment, and it is difficult to know the exact timing of the attack initiation unless the system administrator of the target system is the one who is responsible for the attack.

Thus, the success or failure of the attack depends on matching the timing of the attack pulse transmission to the traffic forwarding period of the attack target, and this makes LDoS attacks on short transfers more difficult than attacks on long transfers.

We aim to explore a new LDoS method that targets short transfers to make clear the attack utilization by the attackers and the damage that could be inflicted by the attack, and the extent to impact could be assessed more precisely for defending. In this paper, we proposed the First-Attack Pulse Width Expansion Shrew (Fawe-Shrew) method for improving timing skew tolerance performance by expanding the initial pulse width and showing the model and conditions for the success the attack.

Our main contributions are twofold:

- We propose the new LDoS attack method named *Fawe-Shrew* method, and formulate the timing-skew tolerance performance.
- We experimentally validate the timing-skew tolerance performance using a test-bed network with actual equipment.

The paper is organized as follows. First, the background and purpose are shown. Section II reviews the related work to highlight existing methods. Section III is an account of the method for increase of timing skew tolerance. Section IV experimentally evaluates timing skew tolerance performance. Finally, Section V concludes the paper.

II. Related Work

In this section, we highlight existing LDoS attack methods, and explain reasons of selecting Shrew method. Then, we introduce shrew strategy and studied on the effectiveness of the Shrew method.

A. Existing LDoS Attack Methods

DoS attacks can be classified into two categories: Flooding DoS (FDoS) and LDoS attacks, which use large and low amounts of traffic, respectively. FDoS attacks are easy to detect because they use large amounts of traffic, while LDoS attacks are stealthy and evade detection methods by using pulse-shaped attack traffic, which lowers the average bandwidth utilization [1].

LDoS attacks against TCP include the Shrew method, which exploits the Retransmission Timeout (RTO) algorithm [2], the Reduction of Quality (RoQ) method, which exploits the Loss-based congestion control algorithm [3], and FB-Shrew, which exploits both RTO and Loss-based congestion control algorithms [5]. These LDoS attack methods use pulsed attack traffic and have low average bandwidth utilization [1].

FB-Shrew exploits both RTO and congestion control algorithms to increase the pulse period compared to the conventional Shrew method, resulting in improved stealth [5, 6]. In reference [4], RoQ method is classified if the pulse period is longer than 5 seconds, and Shrew method is classified if the pulse period is shorter than 5 seconds.

When considering short transfers as attack targets, it can be expected that the longer the pulse period is, the more difficult it is to realize the attack. Therefore, we devise an LDoS attack method for short transfers based on the Shrew method, which has the shortest pulse period.

B. Mechanism of Shrew method

TCP uses a retransmission timer that controls the retransmission process in terms of time, and the expiration of the retransmission timer is called a Retransmission Time Out (RTO) [12]. The retransmission timer management algorithm using exponential backoff has the advantage of being clear and easy to understand, but it also has the vulnerability of predictable retransmission timing.

The initial value of the RTO is defined by RFC 6298 [13] as follows:

$$minRTO = SRTT + \max(G, 4 \times RTTAVR) \qquad (1)$$

In TCP communication, the RTO_n of the n-th RTO is determined by the following equation using exponential backoff:

$$RTO_n = 2 \cdot RTO_{n-1},\ RTO_1 = minRTO \qquad (2)$$

The upper limit of RTO is generally set to 60 seconds. If $minRTO = 1$, a TCP connection timeout occurs when $n > 5$. The Shrew method exploits this timeout mechanism by periodically sending attack traffic every $minRTO$ to continuously generate RTO operations. Although several studies have verified the effectiveness of the Shrew method, all of them focus on long transfers, and none of them consider short transfers of less than one second.

C. Studies on Shrew Method Effectiveness

A typical target of the Shrew method is cloud data center network (DCN).

In cloud computing services, a service provider provides virtual machines as needed by tenants (customers). Computing resources on servers are allocated to the virtual machines, but network resources are shared by tenants. In DCN, the bottleneck-link bandwidth is dynamically changes, thus it is difficult to use a delay-based hop estimation to determine the number of hops of a node path. Therefore, Feng et al. [7] employed a loss-based congestion control algorithm that groups sending virtual machines by flow paths toward the destination. Then, they showed that if the flow rate is higher than switch intermediate buffers, which can congest the buffer, the loss rate also increased monotonically with the number of logical hops in the flow path [7]. This observation can be used to determine which virtual machines reside under the same switch or share the longest node flow path. By using these observations, it becomes even clearer which groups of virtual machines are further from the destination than others. The maximum buffer size available to the switch is synonymous with its capacity to handle burst traffic. Thus, they used the value as the bottleneck link bandwidth.

As a result of the attacking effect of the Shrew method in Cloud Data Center Networks, the Shrew method indicates a TCP throughput reduction of up to 83%. However, the target traffic size is large, and the proposed method takes time to estimate a bottleneck link that changes dynamically [7].

To realize the Shrew method, the transmission rate of the attack pulses must be higher than the bandwidth of the target bottleneck link [2, 10, 14]. Therefore, if the characteristics of the bottleneck link to be attacked are not known, it is difficult to attack successfully. In many cases, attackers don't know the bandwidth of the bottleneck link they are attacking, if the attack rate is insufficient, the attack will not be effective enough, and if the attack rate is too high, the attack will lose stealth. In other words, it is difficult to attack at the attack traffic rate required for a successful LDoS attack.

To solve this problem, Takahashi et al. [8] designed a method that increases the pulse rate in an exploratory, and calculates the ideal attack rate. First, to obtain the bottleneck link bandwidth which is necessary for the attack, construct a BOT node in the target network, and measure the effectiveness of the attack using the BOT node. Next, send attack traffic of pulse rate, which is lower than the bottleneck link bandwidth. Finally, increase the pulse rate until the target attack effectiveness is achieved by using the attack effectiveness observed at the BOT node.

This method makes attacks successful without the bottleneck link bandwidth. However, the process of searching ideal attack pulse rate requires 40 or more seconds. It means this method is not considered the case for the short transfers that we focus on in this study.

In existing studies [2–9], the transfer time of the targeted traffic was of long duration compared to pulse period. The Shrew method generates RTOs continuously by sending $minRTO$ seconds period attack pulses. Therefore, if the timing of the attack pulses cannot be matched with traffic because of transfer time is shorter than the pulse period, RTO will not happen, and the Shrew attack fails. To the best of the authors' knowledge, no research has been reported on LDoS attacks based on the Shrew method, and targets short transfers that can occur in interactive transactions. In this study, we propose a method to improve the timing skew tolerance of the attack initiation timing.

III. First-Attack Pulse Width Expansion Shrew (Fawe-Shrew)

In this study, we propose a First-Attack Pulse Width Expansion Shrew (Fawe-Shrew) method that improves the timing skew tolerance performance of attack timing estimation in order to improve the effectiveness of attacks on short transfers.

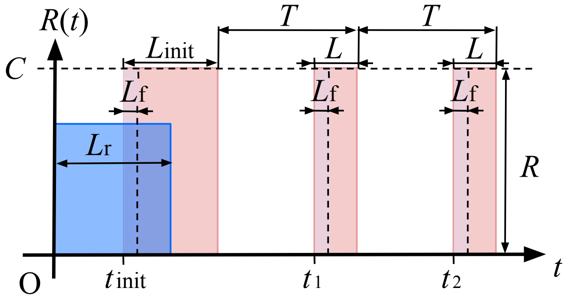


Fig. 2. Traffic model of Fawe-Shrew method

TABLE I
PARAMATERS IN FIG. 2

Meaning	Parameter
attack start time	t_{init} [s]
attack timing by i-th pulse	t_i [s]
transfer duration of regular traffic without attacking	L_{r} [s]
Initial pulse width	L_{init} [s]
buffer fill time	L_{f} [s]
subsequence pulse width	L [s]
pulse period	T [s]
pulse traffic rate	R [Mbps]
bottleneck link bandwidth	C [Mbps]

A. Key Idea of Fawe-Shrew method

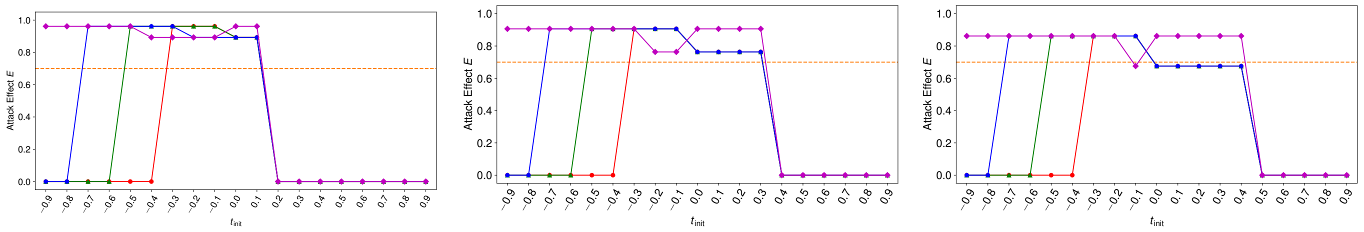
The Fawe-Shrew method generates large traffic only for the first attack to increase the tolerance of timing skew against the attack traffic. By using a large-width initial pulse, even if there is a timing skew in the estimated attack start timing, the attacking traffic will collide with the targeted traffic, and RTO retransmission operations can be caused more reliably. In other words, attacking with Fawe-shrew can be realized without precise timing synchronization of the attack traffic with the target traffic.

After the RTO operation occurs by an initial attack pulse, the Fawe-Shrew periodically sends short attack pulses every $minRTO$, which is the same approach as the conventional Shrew method. Consequently, the Fawe-Shrew method facilitates attack against short transfers while maintaining stealthiness.

B. Relationship of First-Pulse Width and RTO Conditions

Figure 2 shows the model of Fawe-Shrew method, and the meaning of parameter is shown in Table I.

Expanding the first-pulse width is expected to increase the probability of traffic collisions necessary for a successful attack. The conditions under which the attack target traffic



(a) Set $L_{\mathrm{r}} = 0.12$ sec.

(b) Set $L_{\mathrm{r}} = 0.31$ sec.

(c) Set $L_{\mathrm{r}} = 0.48$ sec.

Fig. 3. Calculated E using Equations (3), (4a)–(4c), (5)

will collide with the initial pulse can be expected to be as follows:

$$L - (L_{\text{init}} + L_{\text{f}}) \leq t_{\text{init}} < L_{\text{r}} - L_{\text{f}} \tag{3}$$

From this equation, we can calculate throughput loss rate using targeted traffic transfer time L_{r}, initial RTO timer value $minRTO$, and the number n of RTO happens. We assume that the $minRTO$ value is set to 1 second, which is recommended in [13]. When RTO occurs once during a transfer, throughput decreases to 0.91, 0.67, 0.5 times for transfers with transfer times of 0.1, 0.5, and 1.0, respectively.

However, the subsequent pulses have to also collide with the attack traffic for more reliable throughput reduction. The conditions for the attack traffic to collide with the initial and subsequent pulses, and for two or more RTOs can be predicted as follows:

$$-L_{\text{init}} \leq t_{\text{init}} < L - (L_{\text{init}} + L_{\text{f}}) \tag{4a}$$

$$\begin{aligned} L - (L_{\text{init}} + L_{\text{f}}) \leq t_{\text{init}} < -L_{\text{f}} \\ \wedge\ L_{\text{r}} > t_{\text{init}} + L_{\text{init}} - L + L_{\text{f}} \end{aligned} \tag{4b}$$

$$\begin{aligned} -L_{\text{f}} \leq t_{\text{init}} < L_{\text{r}} - L_{\text{f}} \\ \wedge\ L_{\text{init}} \geq T - L + L_{\text{f}} \end{aligned} \tag{4c}$$

C. *Expected Effectiveness using Fawe-Shrew Model*

The number n of RTOs is expectable using Equations (3) and (4a)–(4c) as the conditions of the n RTOs. We calculate the effectiveness of an attack and draw the attack effectiveness as a function of the first pulse width.

Figure 3 shows an attack effect $E_{\text{calc}}(n)$ as a function of the first pulse width L_{init}. The attack effect $E_{\text{calc}}(n)$ is defined as the rate of throughput decrease when the number of RTOs is n.

$E_{\text{calc}}(n)$ is decrease throughput rate E when the number of RTOs is n. The rate of decrease in throughput which is depending on the number of RTOs is calculated by Equation (5):

$$E_{\text{calc}}(n) = 1 - \frac{L_{\text{r}}}{L_{\text{r}} + minRTO \cdot (2^n - 1)} \tag{5}$$

For creating Figure 3 with the conditions express by Equations (4a)–(4c), we set $n = 2$ because the probability of a traffic collision with the second pulse is very high.

In Figure 3, the range of t_{init} can be expanded in the negative direction to $-L_{\text{init}}$ by expanding the initial pulse width, and the initial pulse width L_{init} should be expanded to $T - L + L_{\text{f}}$ or more even when t_{init} is within the interval $[-L_{\text{f}}, L_{\text{r}} - L_{\text{f}}]$. L_{init} to $T - L + L_{\text{f}}$ or more. Furthermore, when t_{init} is in the interval $[L - (L_{\text{init}} + L_{\text{f}}), -L_{\text{f}})$, if the regular traffic transfer time L_{r} is larger than $t_{\text{init}} + L_{\text{init}} - L + L_{\text{f}}$, we can predict that RTOs will occur two or more times.

IV. Evaluation

To validate the increase of the attack possibility by our proposed Fawe-Shrew method, we evaluated the robustness to attack timing skew on a test-bed network with actual equipment.

A. *Experiment Setup*

Figure 4 shows the network toplogy used in our experiment. We connected a sender and three attackers to a router, which is connected to a receiver via a bottleneck link. The bandwidth of the bottleneck link is set to 60 Mbps, while the bandwidth of the sender side of the router is set to 300 Mbps using Linux tc command. The router transfers all the data on the sender side to the receiver side. An observer is also installed at the receiver side of the router to capture the traffic for evaluation using tcpdump command. The router's queue size is set to 1,000 packets. The attacker sends attack pulses in the direction of the router to occupy the router's queue. The average RTT between sender and receiver under no load is 0.977 ms.

The equipment and protocols used by each entity are shown in Tables II and III, respectively. In this experiment, we use gRPC as an application layer on the sender and receiver, which is used for interactive transactions such as microservice architectures. gRPC uses TCP at the transport layer, so it is possible to generate RTO processing with pulse-shaped attack traffic.

The Fawe-Shrew method was tested with initial pulse widths of 0.5, 0.7, and 1.0 seconds, and compared with the general Shrew method (i.e., with an initial pulse width of 0.3 seconds) to verify the tolerance for each initial attack pulse width.

We changed attack start timing t_{init} and captured traffic data on the observer to calculate attack effectiveness E defined as

$$E = 1 - \frac{P_{\text{onAttack}}}{P_{\text{normal}}} \tag{6}$$

TABLE II
EQUIPMENTS USED AT EACH ENTITIES

Entity	OS	CPU
Sender	Raspberry Pi OS	ARM Cortex-A53
Receiver	Raspberry Pi OS	ARM Cortex-A53
Router	OpenWRT	Intel(R) Celeron(R) J4125 CPU
Attacker	Raspberry Pi OS	ARM Cortex-A53
Observer	Debian	Intel(R) Core(TM) i7-10700

TABLE III
PROTCOLS USED AT EACH ENTITIES

Entity	Network Layer	Transport Layer	Application Layer
Sender	IP	TCP	gRPC
Receiver	IP	TCP	gRPC
Router	IP	-	-
Attacker	IP	UDP	-
Observer	IP	TCP	-

where P_{onAttack} and P_{normal} are throughput with and without attack, respectively. We define the attack is 100% effective ($E = 1.0$) when all data transfer is not completed, i.e., $\sum_{k=1}^{n} RTO_k > 60$ and a session timeout occurs. To represent the timing skew, the attacker node starts sending attack traffic at time $t = t_{\text{init}}$ to the receiver.

The attack pulse rate was set to 0.3 seconds, which is enough to fill the router buffer on 60 Mbps bandwidth. t_{init} was varied from -0.9 to 0.9, for each of the initial pulse widths $L_{\text{init}} = 0.3$, 0.5, 0.7, and 1.0 seconds. To confirm the relationship of transfer time of regular traffic L_{r} we did the above experiment for each 1–3MB data, which is 0.12 seconds when 1MB, 0.32 seconds when 3MB, 0.48 seconds when 5MB. The number of trials was 20 for each condition.

The robustness to attack timing skew is evaluated by using a timing skew tolerance D calculated from the attack effectiveness E as:

$$D = t_{\max} - t_{\min} \tag{7}$$

where $t_{\max}$ and $t_{\min}$ are maximum and minimum t_{init} in the range where E is above a specific threshold E_{th}.

The threshold E_{th} is set to 0.70 based on the maximum allowable wait time for mobile site visitors. Reference [15] shows that the probability of a mobile site visitor bouncing rate increases 32% when the transfer time increases from 1–3 seconds. Based on this fact, we set the maximum allowable transfer time to 3 seconds and calculated the maximum allowable attack effectiveness $E = 0.75$. We set the threshold $E_{\text{th}} = 0.70$ including a margin.

B. Result

Figure 5 shows the average of target attack effective E on the test-bed network with actual equipment for each pulse width concerning t_{init}. We compare Figure 5 with Figure 3 which is created E using Equation (5) with Equations (3), (4a)–(4c). By comparing (a) to (c) of Figure 3 and Figure 5, we can confirm all of $t_{\min}$ are $-L_{\text{r}}$.

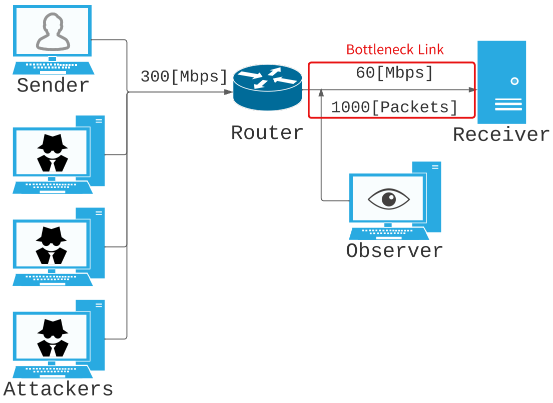


Fig. 4. Test-bed network

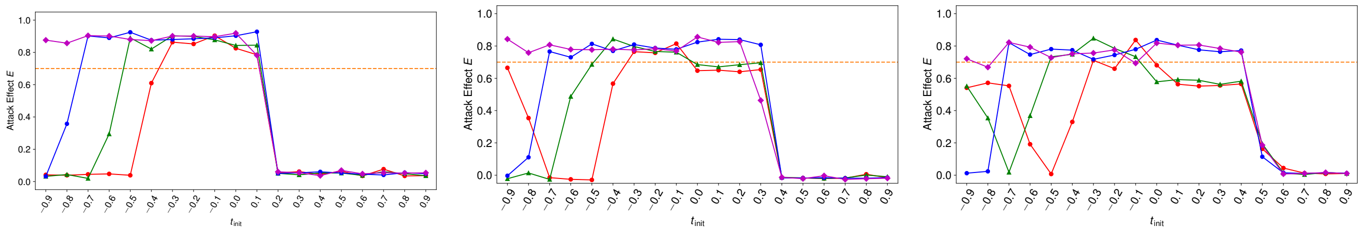
Table IV shows the timing skew tolerance performance D for each pulse width. When we focus on D on test-bed and calculated D_{calc}, the difference $D - D_{\text{calc}}$ is 0 or -0.1 except the condition of 3MB data transfer with $L_{\text{init}} = 0.3, 0.5$, and 5MB data transfer with $L_{\text{init}} = 0.7$.

In the case of 3MB data size and $L_{\text{init}} = 0.3, 0.5$, the expected number of RTO is 1 because of Equation 3. As we said in section III, the attack effectiveness E by 1 RTO of short transfer is higher than long transfer. Because of the difference, the attack effectiveness E by 1 RTO in the condition of 3MB data size and $L_{\text{init}} = 0.3, 0.5$ is close to the threshold $E_{\text{th}} = 0.7$. This makes the difference between D on test-bed and calculated D_{calc}.

In summary, it can be considered that the conditions we showed in Equations E_{calc} with equation (3), (4a)–(4c) is almost correct, and it is inferred that the proposed Fawe-Shrew method is considered to increase the possibility of traffic collision occurrences due to the extended initial attack pulse that the buffer occupies. Thus, the proposed Fawe-Shrew method can improve the attack initiation timing skew tolerance by increasing the initial pulse width.

TABLE IV
TIMING SKEW TOLERANCE PERFORMANCE D

Data size	L_{r}	L_{init}	D on test-bed	calculated D D_{calc}	Difference $(D - D_{\text{calc}})$
1MB	0.12	0.3	0.4	0.4	0
		0.5	0.6	0.6	0
		0.7	0.8	0.8	0
		0.9	1.0	1.0	0
3MB	0.31	0.3	0.2	0.6	-0.4
		0.5	0.4	0.9	-0.5
		0.7	1.0	1.0	0
		0.9	1.1	1.2	-0.1
5MB	0.48	0.3	0.2	0.2	0
		0.5	0.4	0.4	0
		0.7	1.1	0.6	$+0.5$
		0.9	1.3	1.3	0



(a) Data size: 1MB, ave. of $L_{\rm r} = 0.12$ sec.   (b) Data size: 3MB, ave. of $L_{\rm r} = 0.31$ sec.   (c) Data size: 5MB, ave. of $L_{\rm r} = 0.48$ sec.

Fig. 5. Result of test-bed network with actual equipment

C. Discussion

In this study, we proposed the Fawe-Shrew method, which improves the timing skew tolerance of attack initiation timing by expanding the initial pulse width against short transfers compared to the conventional Shrew method. To verify the timing skew tolerance performance of the Fawe-Shrew method, we shifted attack initiation timing. The targeted traffic transfer initiation timing for the four patterns: the conventional Shrew method with initial pulse width $L_{\rm init} = 0.3$ and expanded initial pulse widths $L_{\rm init} = 0.5$, 0.7 and 1.0. We evaluated with an average of attack effectiveness E. The evaluation results show that the proposed Fawe-Shrew method can improve the timing skew tolerance of the attack initiation timing by expanding the initial pulse width. In other words, the Fawe-Shrew method is effective for short transfers by expanding initial attack traffic, and it is needed to indicate further countermeasures. However, in the current Fawe-shrew method, a quantitative and qualitative discussion is needed on the trade-off between pulse-width expansion and the resulting disadvantages. The Shrew method uses pulse-shaped traffic to reduce the attack traffic rate and avoid DDoS attack detection mechanisms that detect large amounts of traffic. However, the Fawe-Shrew method increases the pulse width, so the possibility of being detected by DDoS attack detection mechanisms is due to increasing the amount of traffic. Therefore, it is necessary to verify how much the initial pulse width should be increased to increase the timing skew tolerance D and avoid the detection mechanism.

V. Conclusion

In this study, we inspected the feasibility of the Shrew LDoS attacks method against short transfers. In case of the Shrew method, the attack timing skew has a large impact on the attack effectiveness for short transfers. Thus, we proposed the Fawe-Shrew (First-Attack Pulse Width Expansion Shrew) method to improve the attack effectiveness in the presence of this skew. We confirmed that the proposed method has increased tolerance of the skew in the attack initiation timing. We showed the proposal model is correct especially in short transfer duration, using test-bed network with actual equipment.

Acknowledgment

This work was supported by JSPS KAKENHI Grant Number JP20K11772.

References

[1] W. Zhijun, L. Wenjing, L. Liang *et al.*, "Low-rate dos attacks, detection, defense, and challenges: a survey," *IEEE access*, vol. 8, pp. 43 920–43 943, 2020.

[2] A. Kuzmanovic and E. W. Knightly, "Low-rate tcp-targeted denial of service attacks: the shrew vs. the mice and elephants," in *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, 2003, pp. 75–86.

[3] M. Guirguis, A. Bestavros, and I. Matta, "Exploiting the transients of adaptation for roq attacks on internet resources," in *Proceedings of the 12th IEEE International Conference on Network Protocols, 2004. ICNP 2004.* IEEE, 2004, pp. 184–195.

[4] A. Shevtekar and N. Ansari, "A router-based technique to mitigate reduction of quality (roq) attacks," *Computer Networks*, vol. 52, no. 5, pp. 957–970, 2008.

[5] M. Guirguis, A. Bestavros, and I. Matta, "On the impact of low-rate attacks," in *2006 IEEE International Conference on Communications*, vol. 5. IEEE, 2006, pp. 2316–2321.

[6] M. Yue, M. Wang, and Z. Wu, "Low-high burst: a double potency varying-rtt based full-buffer shrew attack model," *IEEE Transactions on Dependable and Secure Computing*, vol. 18, no. 5, pp. 2285–2300, 2019.

[7] Z. Feng, B. Bai, B. Zhao *et al.*, "Shrew attack in cloud data center networks," in *2011 Seventh International Conference on Mobile Ad-hoc and Sensor Networks*. IEEE, 2011, pp. 441–445.

[8] Y. Takahashi, H. Inamura, and Y. Nakamura, "A low-rate ddos strategy for unknown bottleneck link characteristics," in *2021 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events (PerCom Workshops)*. IEEE, 2021, pp. 508–513.

[9] G. Maciá-Fernández, J. E. Díaz-Verdejo, and P. García-Teodoro, "Evaluation of a low-rate dos attack against iterative servers," *Computer Networks*, vol. 51, no. 4, pp. 1013–1030, 2007.

[10] J. Luo, X. Yang, J. Wang *et al.*, "On a mathematical model for low-rate shrew ddos," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 7, pp. 1069–1083, 2014.

[11] A. K. Baitha and S. Vinod, "Session hijacking and prevention technique," *Int. J. Eng. Technol*, vol. 7, no. 2.6, pp. 193–198, 2018.

[12] W. Eddy, "Rfc 9293: Transmission control protocol (tcp)," 2022.

[13] V. Paxson, M. Allman, J. Chu *et al.*, "Rfc 6298: Computing tcp's retransmission timer," 2011.

[14] S. Sarat and A. Terzis, "On the effect of router buffer sizes on low-rate denial of service attacks," in *Proceedings. 14th International Conference on Computer Communications and Networks, 2005. ICCCN 2005.* IEEE, 2005, pp. 281–286.

[15] D. An, "Find out how you stack up to new industry benchmarks for mobile page speed," *Think with Google-Mobile, Data & Measurement*, p. 24, 2018.