CACHED GUARANTEED-TIMER RANDOM DROP AGAINST TCP SYN-FLOOD ATTACKS AND FLASH CROWDS

Hiroshi Fujinoki
Department of Computer Science
Southern Illinois University Edwardsville
Edwardsville, Illinois 62026-1656 USA
E-mail:hfujino@siue.edu

ABSTRACT
This paper presents a new method for improving web server performance and fairness in the face of SYN-flooding and flash crowds. The method proposes use of cache to avoid preemption of legitimate SYN messages from the TCP backlog queue in Random Drop (RD) method. A new algorithm, the Cached Guaranteed Timer Random Drop (Cached GT-RD), was designed to maximize the effect of the cache during flash crowds. Performance of the Cached GT-RD was evaluated and compared to an existing solution, the Probabilistic Pre-filtering Random Drop (PP-RD), using the simulation method. The experiments demonstrated that Cached GT-RD improved the connection rate and throughput by 67.4 and 73.2% from PP-RD. Cached GT-RD also improved the fairness for slow-connection clients, who most suffer from SYN-flooding attacks and flash crowds. For small TCP backlog queue, the successful connection rate of slow-connection clients became four times better than PP-RD. The proposed solution does not require any modification in either hardware or software for existing data transmissions using TCP/IP. The results of simulation experiments suggest that use of cache will be an efficient and practical solution for both SYN-flooding attacks and flash crowds and Cached GT-RD will be effective in improving fairness in connections.

KEY WORDS
Network security, denial-of-service, and flash crowd

1. INTRODUCTION

Keeping a web server up to public access is one of the most significant issues especially for corporate web-site owners. Regarding this, a type of security attacks, called denial-of-service attacks, have been recently a serious problem [1]. In denial-of-service attacks, malicious users perform attacking activities to disable servers from their normal operations. We call such malicious users "attackers" to distinguish them from legitimate users, who access a server without intention of such attacks.

Denial-of-service attacks (abbreviated as "DoS attacks" hereafter) are classified into two categories of logic attacks and flooding attacks [2]. Logic attacks are those that exploit weaknesses in software implementation of transport, network, and routing protocols to prevent a server from functioning (or from running at its full performance). In flooding attacks, attackers dump a large volume of traffic to a target server, trying to let a server host exhaust resources to process requests from legitimate clients.

Although flooding attacks can happen in many different ways, the most popular is TCP SYN-flooding attack [3, 4, 5]. Under a SYN-flooding attack, SYN requests from an attacker will never leave the TCP backlog queue until one expires (after 75 seconds in TCP's default setting). Any new SYN message from legitimate clients will be rejected while the backlog queue is full. Attackers can block the TCP backlog queue with a small number of SYN messages since the default queue size in most of the commercial operating systems is small (between five to ten slots [5]).

The existing solutions against SYN-flooding attacks are classified as attack detections [6, 7], connection establishment improvements [8, 9], back-tracing [10, 11], limiting attacking SYN rate [12], and dropping SYN packets with a spoofed source address [13], any of which has not been an ultimate fix for different reasons. Surveys of existing solutions for SYN-flooding attacks can be found in [5, 14].

What makes defense against SYN-flooding attacks difficult is the situation, called a flash crowd. A flash crowd is a large volume of legitimate access requests from all legitimate clients in a sense that the sheer volume of requests overwhelms a server [15, 16]. The primary cause of the performance degradation during a flash crowd is thrashing, named "receive live-lock" by Mogul [17]. During a receive live-lock, the server throughput could drop even close to zero once a server hits such a situation.

In this project, we focused on a protection of web servers that tries to let web servers to continue to operate during both a TCP SYN-flooding attack and a flash crowd. The rest of this paper is organized as follows. Section 2

describes existing solutions, Random Drop (RD) and Probabilistic Pre-filtering Random Drop (PP-RD) algorithms. Section 2 analyzes their advantages and disadvantages. Section 3 introduces our new and efficient packet drop algorithm, Cached Guaranteed Timer Random Drop (Cached GT-RD). Section 4 presents performance evaluation using simulation experiments. Section 5 provides conclusions of this project and its future work, followed by a list of references.

2. EXISTING SOLUTIONS

Random Drop (RD): Random Drop (RD) protects a server from TCP SYN-flooding attacks by randomly dropping one of the pending SYN requests in the TCP backlog queue [18, 19]. Flow chart (a) in Figure 1 shows the procedure in RD. When a new SYN message arrives at the TCP layer in a server host, the TCP backlog queue will be scanned to find an empty slot (marked as ① in the figure). If an empty slot is found, the new SYN message is placed in an empty slot (②). Then, SYN-ACK message is transmitted to the connecting client (③). When the reply for the SYN-ACK message arrives at the server host (④), a TCP connection is established. If a reply does not come back within 75 seconds, the SYN message will be dropped from the queue (⑤). If the backlog queue is full, one of its pending SYN requests will be randomly dropped (⑥) to make a space for the new SYN message. The rest is continued from ②.

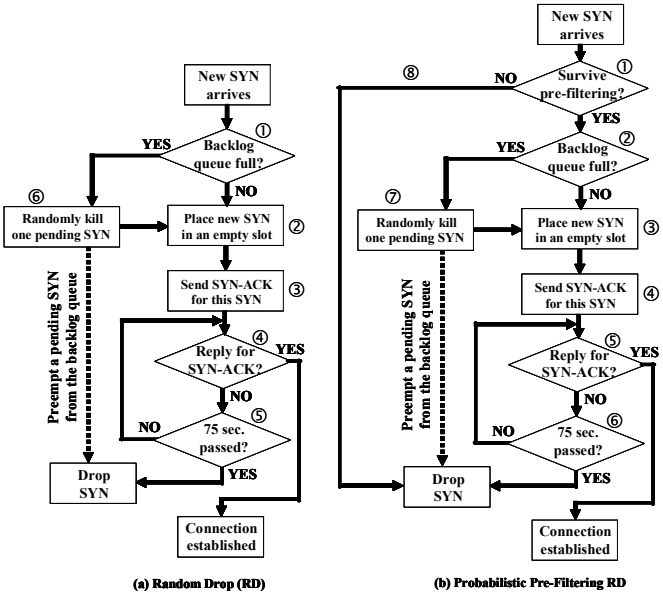

Figure 1 - Procedure for RD (a) and PP-RD (b)

The primary advantage in RD is to prohibit attacking SYN messages from clogging the TCP backlog queue, so that legitimate SYN messages can always get a slot in the backlog queue. RD, however, has two inefficiency problems. The first problem is thrashing during a flash crowd. If a large number of SYN messages are delivered to a server, RD will not give enough time even for legitimate clients to respond to SYN-ACK messages from a server. This is because a large number of incoming SYN messages will continuously preempt those for legitimate clients already in the backlog queue.

Another problem in RD is unfairness against legitimate clients using a slow connection, such as analog modem users. Due to long transmission and buffering delay for slow connections, the response time (the delay after a server sends SYN-ACK message until a server receives a reply for it) of such slow-connection clients is usually longer than the clients using high speed connections [5, 18]. This implies that such slow-connection clients will always be the victims in RD (since slow-connection clients require longer time to complete the procedure of the TCP three-way handshaking, they will have high probability of being preempted before they establish a TCP connection).

Probabilistic Pre-filtering Random Drop (PP-RD): Ricciulli proposed a solution for the thrashing and starvation problem in RD [14]. Ricciulli's solution is called Probabilistic Pre-filtering Random Drop (PP-RD) in this work. The solution tries to solve the problem by using a probabilistic pre-filtering in front of the TCP backlog queue. The procedure of PP-RD is shown in (b) in Figure 1. When a new SYN message reaches the TCP layer at the server, the pre-filter will randomly drop it at the probability of (1-*K*) (①), where *K* is the survival rate ($0 < K \leq 1.0$). The incoming SYN messages that do not survive the pre-filtering will be dropped before they reach the TCP backlog queue (⑧). The survived SYN messages follow exactly the same procedure defined in RD (② through ⑦).

To efficiently avoid preempting legitimate SYN messages pending in the TCP backlog queue, the survival rate is dynamically adjusted using the following formula.

$$K = \frac{-1}{\ln(1-1/q)\cdot(R_{good}+R_{bad})\cdot T} \qquad (1)$$

where R_{good} is the request rate (i.e., number of SYN messages arrived in a second) by legitimate clients and R_{bad} is that of attacker(s). The parameter, q, represents the number of slots in the TCP backlog queue while T is the average response time to the SYN-ACK message by legitimate clients. By controlling the number of SYN messages that will reach the TCP backlog queue, preemption of legitimate SYN messages can be decreased.

As a possible problem in PP-RD, if the ratio of R_{bad} to R_{good} increases, the pre-filtering may drop most of the SYN messages from legitimate clients. This is because PP-RD does not distinguish legitimate requests from attacking requests. If a large number of requests are blindly dropped, this will drop the majority of a relatively small number of legitimate requests (i.e., pre-filtering will not increase the population ratio of the legitimate requests) to result in a low connection rate for legitimate clients during attacks.

3. PROPOSED SOLUTION

The proposed solution is named Cached Guaranteed Timer Random Drop (Cached GT-RD) and it also aims for fair service for legitimate clients using slow connections. It introduces the guaranteed timer in addition to the hard expiration timer already used in the TCP (i.e., the 75-second expiration timer).

The guaranteed timer guarantees a new SYN request to stay at the TCP backlog queue (i.e., will not be preempted) for a designated time interval. The purpose of the guaranteed timer is to prevent legitimate SYN messages in the TCP backlog queue from being preempted by giving them enough time to respond to SYN-ACK even under a flash crowd or a SYN-flooding attack. Each slot in the TCP backlog queue is assigned the guaranteed timer, which should be shorter than the TCP's hard timer (i.e., the 75-second timer). The same value of the guaranteed timer can be assigned to all the slots in the backlog queue, or each slot can be assigned a different value.

In web traffic, it is expected that multiple connections will be established in a short time following the first connection request from a client. The first connection will download the definition of a page (such as "index.html"), in which links to multiple files that construct the page exist. This means that downloading the first file in a web page will trigger multiple TCP connections.

To take advantage of this access pattern in web traffic, cache for successfully established connections was combined with the guaranteed timer. When a client successfully establishes the first connection, the client information (the address and port number) will be stored in the cache to give the highest priority to the subsequent connection requests from the same client. Entries in the cache will be replaced based on the least recently used algorithm. Note that attackers can never utilize the cache because SYN-flooding attackers never establish a connection. As long as client entries survive in the cache for long enough to complete downloading a set of files that construct a web page, it will significantly improve the efficiency by eliminating unwanted preemptions.

Figure 2 shows the procedure of Cached GT-RD algorithm. If the backlog queue is full, the backlog queue is searched for any cache-miss SYN message (①). Cache-miss SYN messages are those whose matching entry is not found in the cache when they arrive at a server, but still accepted in the backlog queue. If no cache-miss SYN message exists in the backlog queue (i.e., if all SYN messages in the backlog queue are cache-hit), a new SYN message is immediately dropped (②). If at least one cache-miss SYN is found pending in the backlog queue, then a matching entry for the new SYN message is searched in the cache (③). If a matching entry is found, the new SYN message becomes cache-hit. Otherwise, it becomes a cache-miss SYN.

If the new SYN becomes a cache-miss SYN, the backlog queue is searched for the pending SYN messages that are cache-miss (④) and whose guaranteed timer has expired (⑤). If such SYN messages exist in the backlog queue, Cached GT-RD randomly drops one and replaces it by the new SYN (⑥). If not, the new SYN message is dropped immediately (⑦).

If the new SYN becomes cache-hit, then the backlog queue is searched for cache-miss pending SYN messages whose guaranteed timer has already expired (⑧). The difference of ⑧ from ⑤ is that the new cache-hit SYN will be accepted no matter if any pending SYN message whose guaranteed timer has expired or not (⑨). If there are some pending SYNs whose guaranteed timer has expired, one of them will be randomly replaced by the new cache-hit SYN (if this is the case, any of the other pending SYNs whose guaranteed timer has not expired will not be preempted). If not, any one of the pending SYNs in the backlog queue will be randomly replaced by the new cache-hit SYN (⑨). If the new SYN message is accepted by the TCP backlog queue, the procedure of TCP three-way handshaking starts (⑩).

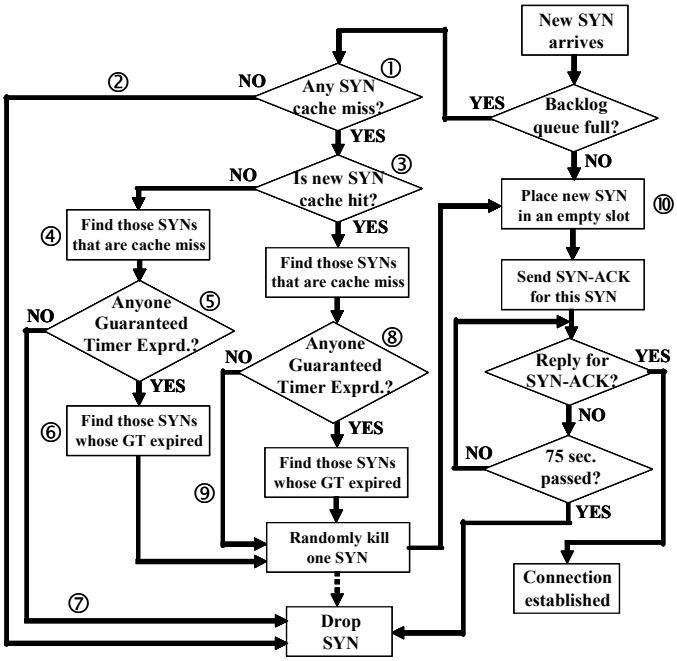

Figure 2 – Procedure for Cached GT-RD

4. PERFORMANCE EVALUATION

Simulation experiments were performed to evaluate the performance of Cached GT-RD and to compare its performance to PP-RD. This section describes modeling of the experiments, experiment designs and observed results from the experiments.

4.1 *Experiment Modeling*

The simulation experiments measured the rate of successful connections for legitimate clients, the number of unsuccessful SYN requests and preempted SYN

requests for attacking and legitimate clients under a SYN-flooding attack and a flash crowd. The TCP backlog queue was modeled by a one-dimensional array of N slots. Each SYN message carried the following three parameters: class of a client, expected response time, and time stamp of arrival at the TCP layer. The three parameters are defined below.

Class of a client: Four different classes of clients were simulated in the experiments. Class-F clients modeled the legitimate clients using high-speed connections. Class-S clients were the legitimate clients using slow and high-latency connections. Class-G clients were the legitimate clients that went through a congested network and Class-A clients modeled SYN-flooding attackers. These four different classes of clients were simulated by different average response times (defined in the next paragraph) to SYN-ACK message transmitted by the server TCP.

Average response time: The expected delay after the SYN-ACK message is transmitted to a connecting client but before the client's reply to the SYN-ACK message comes back to a server. Therefore, the response time simulated the length of time a SYN message occupied a slot in the backlog queue. For class-A SYN messages, the average response time was always infinity (i.e., the reply for a SYN-ACK message never came back to a server). For legitimate clients using a high-speed connection (class-F), the average response time is expected to be shorter than that of those using a slow or a congested connection (class S or G). The response time for each SYN message was modeled by Bounded Pareto distribution to simulate a long tail in their distribution.

Time stamp of arrival: The time stamp of a SYN message's arrival at the TCP backlog queue. The inter-arrival times of arriving SYN messages were modeled by Poisson distribution. The arrival rate of SYN messages was controlled by the average SYN inter-arrival time.

Other major parameters included, attack rate and population mix of legitimate clients. The attack rate was the ratio of the number of SYN messages submitted by attackers to that by legitimate clients. The population mix of legitimate clients was the ratio of the number of SYN messages submitted by all the legitimate clients.

4.2 *Experiment Designs*

To compare the performance of Cached GT-RD to PP-RD algorithm, the following three experiments were designed and they are defined below.

Experiment #1 (SYN Rate Experiments): The effect of the SYN arrival rate to the successful connection rate was studied. The successful connection rate (called "connection rate" hereafter) was defined as the ratio of the number of successfully established TCP connections to the total number of SYN messages submitted by all legitimate clients. SYN rate was controlled by the average SYN inter-arrival time (denoted by "$T_{SYN\text{-}int}$"). Various SYN rates from 100 SYNs/second (whose $T_{SYN\text{-}int}$ was 10ms) to 10,000 SYNs/second ($T_{SYN\text{-}int}$ was 0.1ms) were tested for PP-RD and Cached GT-RD.

In Experiment #1, connection throughput, dropped SYN messages, preempted messages and connection rate for each of the three different classes of legitimate clients were also measured. Dropped SYN messages were those that were dropped without having a chance to be in the TCP backlog queue while the preempted SYN messages were those that were once placed in the queue but preempted by another SYN message or its TCP's hard timer expired.

For other parameters, The TCP SYN timeout was set to 75 seconds (TCP's default value). The shape parameter, the upper bound and the lower bound for Bounded Pareto distribution were set to 0.8, 40ms and 300ms, respectively. To model long response times for class-S and G clients, the same Bounded Pareto distribution was used to generate random numbers and then they were multiplied by 3.0 and 8.0 for class-S and G clients. The population mix of the clients was 20, 20, 20 and 40% for Class-F, S, G and A clients, respectively. The interval of the guaranteed timer was set 200ms for all the slots in the backlog queue.

To evaluate the effect of the cache, we implemented GT-RD (Guaranteed Timer Random Drop), which was the Cached GT-RD without the cache (the operations ① through ⑦ in Figure 2 were removed). The same experiments were performed for GT-RD to evaluate the impact of the cache in Cached GT-RD algorithm.

Experiment #2 (Size of Backlog Queue Experiments): The effects of TCP backlog queue size (in number of slots in the backlog queue) to the connection rate were studied. Connection rates of Cached TG-RD algorithms under various backlog queue sizes were measured and compared to PP-RD algorithm. $T_{SYN\text{-}int}$ = 5ms was applied in this experiment. All the other parameters remained unchanged from Experiment #1.

Experiment #3 (Hit Rate Experiments): It was expected that the cache hit rate in Cached GT-RD would have significant impact to its performance. To study how the cache hit rate affected the connection rate in Cached GT-RD, the connection rate was measured for various cache hit factors of 0.1 through 10. The cache hit factor was the cache hit rate in terms of the number of SYN messages that would result in a cache hit for each initial successful connection. For example, cache hit factor of 0.1 means that only one out of ten initial successful connections will make one subsequent SYN message cache hit, while 0.5 means that two out of ten initial successful connections will result in one subsequent SYN cache hit. Hit factor of 2 means that every initial SYN message will trigger two cache hits in the subsequent SYN messages from the same client.

4.3 *Analysis for the experiment outcomes*

This section describes and analyzes the outputs from the experiments. Figure 3 shows the connection rates and connection throughputs for the three algorithms for

various SYN rates. Connection rates of the three algorithms are shown in percentage by the three bar graphs in the figure (their index is the left Y-axis). Connection throughputs are shown by the line graphs in the average number of connections per second (the right Y-axis).

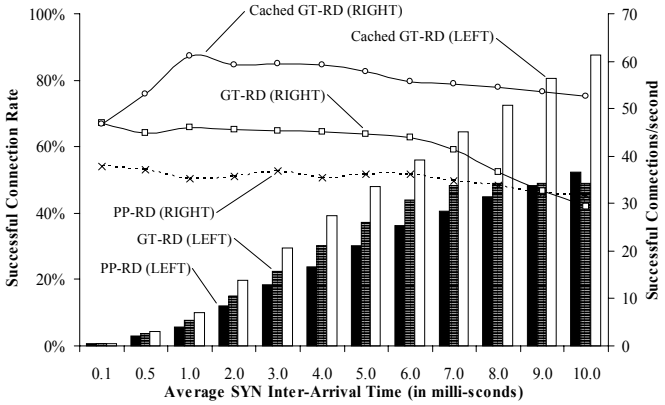

Figure 3 – Connection rate and connection throughputs for PP-RD, GT-RD and Cached GT-RD

The line graphs show that the connection throughput of Cached GT-RD was constantly better than PP-RD by more than 50% (calculated as ((throughput of Cached GT-RD)-(throughput of PP-RD)/(throughput of PP-RD), except for $T_{SYN\text{-}int}$ = 0.1 and 0.5ms. Cached GT-RD was better than PP-RD by 23.6% at $T_{SYN\text{-}int}$ = 0.1ms, where the throughput of Cached GT-RD was 46.6 connections per second, while that of PP-RD was 37.9. At $T_{SYN\text{-}int}$ = 1ms, the largest improvement of 73.2% was observed. For $T_{SYN\text{-}int}$ = 2 through 10ms, Cached GT-RD improved the connection throughput of PP-RD by 53.2 to 65.7%.

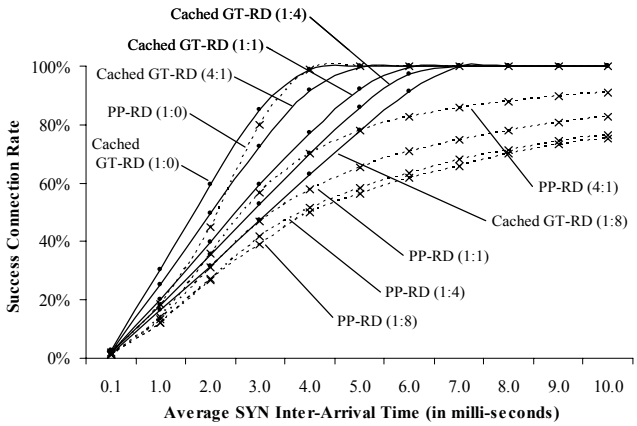

Figure 4 – Successful connection rate for various R_{Good}: R_{Bad} ratio for the three algorithms

The connection rates for GT-RD stopped improving after $T_{SYN\text{-}int}$ = 7ms, while that of Cached GT-RD still continued to improve. These results demonstrated the effect of the cache, since the only primary difference between GT-RD and Cached GT is cache. At $T_{SYN\text{-}int}$ = 10ms, they were 52.4, 48.9 and 87.7% for PP-RD, GT-RD and Cached GT-RD. The largest relative improvement of 67.4% (calculated as (87.7-52.4)/52.4 = 0.674) was observed between Cached GT-RD and PP-RD at $T_{SYN\text{-}int}$ = 10ms.

Figure 4 shows the connection rates for PP-RD and Cached GT-RD for five different attack rates. The ratio of 1:0 means a flash crowd while 1:4 means a SYN-flooding attack with four attacking SYN messages in every five new arriving SYN messages. The connection rates for PP-RD were 1.8, 18.3, 45.1 and 80.0% for $T_{SYN\text{-}int}$ = 0.1 through 3.0ms, while they were 2.9, 30.2, 59.4 and 84.9% for Cached GT-RD. After $T_{SYN\text{-}int}$ = 4ms, they were all 100% for both PP-RD and Cached GT-RD.

For other than 1:0 rate, the growth rate in the connection rate for PP-RD was slower than that of Cached GT-RD especially after $T_{SYN\text{-}int}$ = 3ms. At $T_{SYN\text{-}int}$ = 6ms, Cached GT-RD was better than PP-RD by 21.0% (for 4:1), 40.0% (for 1:1), 52.7% (for 1:4) and 47.8% (for 1:8). These results implied that Cached GT-RD was efficient as a protection against SYN-flooding attacks.

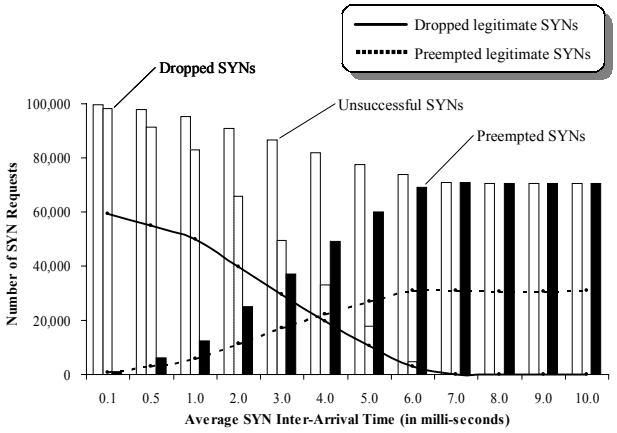

Figure 5 – Number of unsuccessful, dropped and pre-empted SYN messages for PP-RD

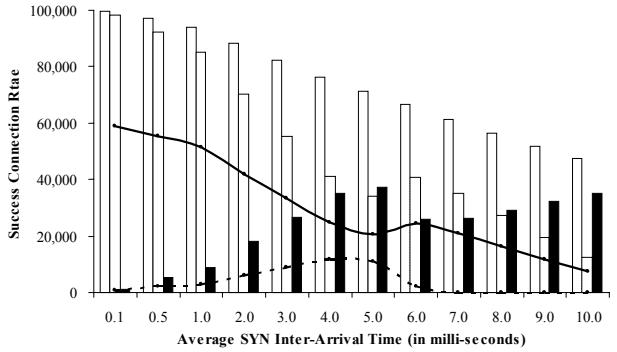

Figure 6 – Number of unsuccessful, dropped and pre-empted SYN messages for Cached GT-RD

Figures 5 and 6 show the number of unsuccessful, dropped and preempted SYN messages when the attack rate was 3:2. Unsuccessful SYN messages were those either dropped or preempted (thus, the sum of the dropped and preempted SYN messages always matched to the number of unsuccessful SYN messages). Bar graphs in the figures show the numbers of dropped and preempted

SYN messages for both legitimate and attacking SYN messages while the line graphs show those only for legitimate clients.

What was common in the three algorithms is that the number of dropped SYNs (as shown by horizontal-stripe bars) and unsuccessful SYNs (white bars) were almost same at high SYN rates ($T_{SYN\text{-}int}$ = 0.1 through 1.0ms). This means that most of the SYN messages were dropped before they were placed in the TCP backlog queue. Although the test case included traffic from attacking clients, this indicated a situation of a flash crowd.

For high SYN rates of $T_{SYN\text{-}int}$ = 0.1 through 1.0ms, the drop rate of SYN messages from legitimate clients (shown by the solid line) was also high. The population ratio of legitimate to attacking SYN messages was 3:2, which means that 60,000 of 100,000 total SYN messages were from legitimate clients. In the three algorithms, the number of dropped SYN messages from legitimate clients was close to 60,000, indicating that most of the SYN messages from legitimate clients were dropped before they were placed in the TCP backlog queue. A large number of dropped SYN messages in Cached GT-RD must have been due to lack of empty slots when they arrived at the backlog queue. For this situation, the most effective way to improve connection rate would be to increase the TCP backlog queue size.

As the SYN rate slowed down ($T_{SYN\text{-}int}$ = 2 through 6ms), the number of preempted SYN messages was observed to start increasing at the same time the number of dropped SYN messages decreased for the three algorithms. After $T_{SYN\text{-}int}$ = 6ms, majority of the unsuccessful SYNs matched with those of the preempted SYNs (black bars) for PP-RD and GT-RD, which contributed to the inefficiency in PP-RD and GT-RD. When the SYN rate was high ($T_{SYN\text{-}int}$ = 0.1 through 3.0ms), the backlog queue was overwhelmed by the sheer volume of arriving traffic, while many of the SYN messages were received just to be preempted at low SYN rates of $T_{SYN\text{-}int}$ = 4 through 10ms in PP-RD and GT-RD (the results of GT-RD are not shown since they were similar to the results of PP-RD).

In Cached GT-RD, the number of preempted SYN messages was limited to be low or zero especially after $T_{SYN\text{-}int}$ = 6ms. Although the number of dropped legitimate SYN messages was higher than that in PP-RD and GT-RD, Cached GT-RD avoided preempting already-received legitimate SYN messages. Since there was no legitimate SYN messages preempted after $T_{SYN\text{-}int}$ = 6ms, the existence of dropped legitimate SYN messages must have been due to lack of capacity in the backlog queue in Cached GT-RD.

The better connection throughput and connection rate in Cached GT-RD can be explained in the following way. In Cached GT-RD, cache-hit SYN messages will never be preempted, which will avoid preemption of legitimate SYN messages once they become cache-hit. On the other hand, SYN messages from attacking clients will always result in cache-misses (this is because attacking clients will never establish a connection). As a result, Cached GT-RD let cache-hit legitimate clients to drop or preempt attacking SYN messages, but not vice versa. However, legitimate clients need to get through their first connections to take the advantage of the cache. The guaranteed timer helps legitimate clients to make their first connection.

Figure 7 shows the connection rates and throughputs for various TCP backlog queue sizes (Experiment #2). The line graphs show the connection rates (their index is the left Y-axis). The bar graphs show the connection throughputs in number of connections per second (the right Y-axis).

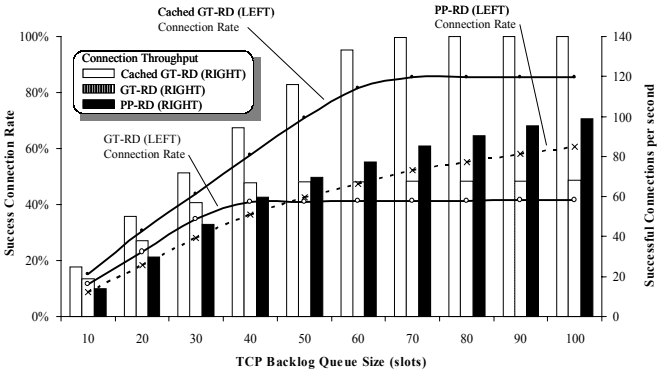

Figure 7 – Effect of the TCP backlog queue size for connection rate and throughput for the three algorithms

Results of this experiment imply that PP-RD is less efficient than Cached GT-RD in utilizing extended capacity in the TCP backlog queue. The connection throughput in Cached GT-RD linearly increased up to 119.6 connections per second (it reached 100% connection rate) at 70 slots. The connection throughput was 57.8 and 73.1 for GT-RD, and PP-RD at 70 slots. Their connection rate for GT-RD, PP-RD and Cached GT-RD was 48.2, 61.0, and 99.8%. PP-RD resulted in a parabola curve, implying that as the backlog queue size increases, the ratio of improvement in connection throughput continued to slow down.

Connection throughput of GT-RD stopped improving at queue size of 40 slots. That was most probably because GT-RD allowed even attacking SYN messages to occupy slots at least for the interval of the guaranteed timer (as a result, connection throughput was equal to simulation time divided by the interval of the guaranteed timer). Cached GT-RD allows cache-hit legitimate SYN messages to preempt attacking SYN messages. Cached GT-RD resulted in more than 50% of improvement in connection rate (as absolute increase) over GT-RD (which was more than 100% increase relative to the connection rate of GT-RD).

Figure 8 shows the connection rates of the three classes of legitimate clients for different TCP backlog queue sizes. The results of this experiment brought two observations. (1) Cached GT-RD was efficient in utilizing increases in the backlog queue size. (2) Except for a small backlog queue size of 10 and 20 slots (where a

flash crowd was most probably happening due to lack of backlog queue slots), Cached GT-RD provided fair opportunities for connections for all three classes of legitimate clients.

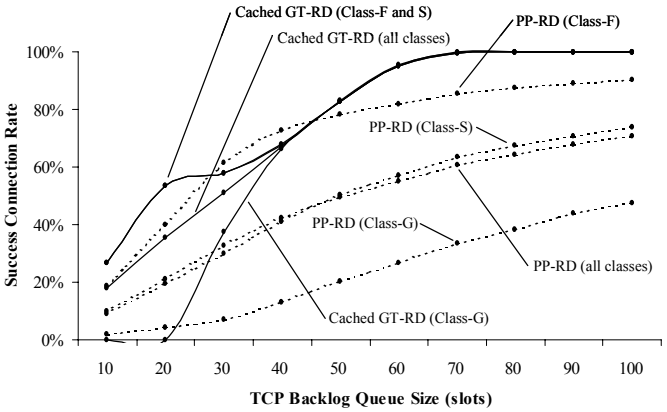

Figure 8 – Effect of the TCP backlog queue size to the success connection rate for the three different classes of legitimate clients for PP-RD and Cached GT-RD

For observation (1), the growth rate in connection rate was much lower in PP-RD than that of Cached GT-RD (the mean growth in PP-RD and Cached GT-RD were 9.0 and 15.5% for the queue size of 10 through 60). In PP-RD, the connection rates of the three classes of clients resulted in almost parallel curves, implying that the connection rate of class-S and G may not converge on class-F without having a large number of additional backlog queue slots.

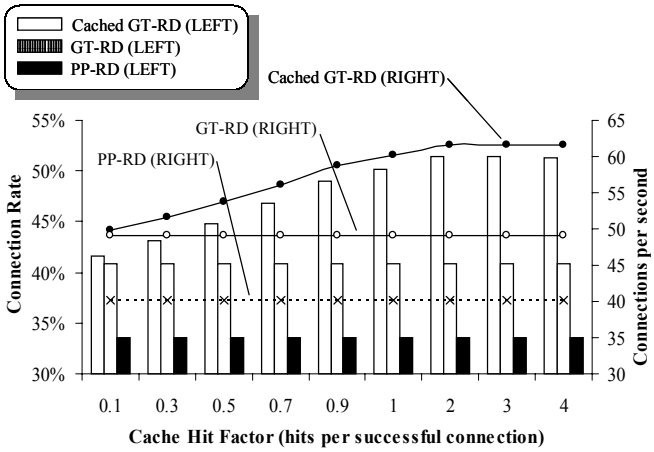

Figure 9 – Effect of cache hit rate to connection rate and throughput

Figure 9 shows the connection rate (as the bar graphs) and throughput (as the line graphs) observed in Experiment #3. At the cache hit factor of 2 or more, Cached GT-RD resulted in 53.2 and 25.7 % improvement in throughput over PP-RD and GT-RD (the throughput of Cached GT-RD was 61.6 connections per second, while they were 40.2 and 49.0 for PP-RD and GT-RD). The connection rate of Cached GT-RD (51.4%) was improved from PP-RD (33.5%) and GT-RD (40.9%) by the same rate (53.4 and 25.7% from PP-RD and GT-RD).

The results of Experiment #3 demonstrate feasibility in Cached GT-RD algorithm. Given a fact that the average number of files requested from within a web page is at least 10 for most web sites [20], hit factor of 2 (two files out of ten file accesses should be cache hit) will not be a number difficult to achieve. For low cache factor of 0.1 through 0.5, Cached GT-RD still resulted in better connection rate and throughput than PP-RD by at least 24%. This implies that Cached GT-RD will be effective not only for web servers but for general servers.

5. CONCLUSION

A new algorithm, Cached Guaranteed Timer Random Drop (Cached GT-RD), for protecting web servers from SYN-flooding attacks and flash crowds was proposed and evaluated in this project. The successful connection rate and connection throughput of Cached GT-RD were evaluated and compared to Probabilistic Pre-filtering Random Drop (PP-RD) using the simulation method.

Cached GT-RD combined two techniques, the guaranteed timer and caching for previously-succeeded clients. The guaranteed timer is to provide enough time for legitimate clients to respond to SYN-ACK message from a server, which helps new clients make their first connections during flash crowds. Caching previously succeeded connections, on the other hand, is to give the highest priority to those clients who have been confirmed for their identity and to avoid preempting legitimate SYN messages already in the backlog queue during SYN-flooding attacks.

Simulation experiments demonstrated that Cached GT-RD resulted in a better connection throughput than PP-RD by 53.2 to 73.2% for most of the SYN rates tested. At $T_{SYN\text{-}int}$ = 1ms, throughput of Cached GT-RD was 60.7 connections per second while that for PP-RD was 34.1 (Figure 3). Cached GT-RD was more efficient than PP-RD in preventing SYN-flooding attacks when $T_{SYN\text{-}int}$ was more than 3ms. For example, at $T_{SYN\text{-}int}$ = 6ms, Cached GT-RD was better than PP-RD by 21.0% (for 4:1), 40.0% (for 1:1), 52.7% (for 1:4) and 47.8% (for 1:8) (Figure 4).

Although the connection rate of Cached GT-RD was low (0.78%) for a high SYN rate ($T_{SYN\text{-}int}$ = 0.1ms), the primary cause of the low connection rate was due to lack of capacity in the TCP backlog queue (as shown in Figures 5 and 6). However, Cached GT-RD still resulted in the highest connection rate in the three algorithms. Therefore, the results of Experiment #2 showed that Cached GT-RD will take more benefit in extending the size of TCP backlog queue than PP-RD.

In PP-RD and GT-RD, when the SYN rate was high, the backlog queue was overwhelmed by the sheer volume of arriving traffic, while many of the SYN messages were received just to be preempted at low SYN rates in PP-RD and GT-RD (Figure 5). Cached GT-RD improved the connection rate by reducing preemptions of legitimate SYN messages in the TCP backlog queue (Figure 6).

For the backlog queue with more than 30 slots, the connection rate of the slowest-connection (class-G) clients was improved by 109.8 (for 100 slots) to 420.8% (30 slots), relative to the connection rate of PP-RD (Experiment #2). For more than 40 slots, the connection rate of class-G clients converged on those for faster client classes (Figure 8). Results of Experiment #3 demonstrated that Cached GT-RD will be a promising technique that will improve connection rate and throughput by reasonably low hit factor (Figure 9). Another advantage in Cached GT-RD is that it does not require any extra hardware or any modification to the existing TCP. Cached GT-RD is transparent to the current implementations of TCP.

REFERENCES:

[1] CERT Coordination Center, Denial of Service Attacks, URL: *www.cert.org/tech_tips/denial_of_service.html*, June 2001.

[2] A. Hussain, J. Heidemann, and C. Papadopoulos, A Framework for Classifying Denial of Service Attacks, *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Karlsruhe, Germany, 2003, 99-110.

[3] S. Bellovin, Security Problems in TCP/IP Protocol Suite, *Computer Communication Review*, *19*(2), 1989, 32-48.

[4] K. J. Houle and G. M. Weaver, Trends in Denial of Service Attack Technology, *Technical Report v1.0*, CERT Coordination Center, 2001.

[5] C. L. Schuba, I. Krsul, M. Kuhn, E. Spafford, A. Sundaram, and D. Zamboni, Analysis of Denial of Service Attacks on TCP, *Proceedings of the IEEE Symposium on Security and Privacy*, Oakland, CA, 1997, 208-223.

[6] H. Wang, D. Zhang, and K. Shin, Detecting SYN Flooding Attacks, *Proceedings of the IEEE INFOCOM*, New York City, NY, 2002, 1530-1539.

[7] A. Habib, M. M. Hafeeda, and B. K. Bhargava, Detecting Service Violations and DoS Attacks, *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Karlsruhe, Germany, 2003, 177-189.

[8] E. Shenk, Another new thought on dealing with SYN flooding, URL: *http://www.wcug.mmu.edu/lists/netdev/199609/msg00171.html*, September 1997.

[9] D. J. Bernstein, SYN Cookies, URL: *http://cr.yp.to/syncookies.html*, September 1996.

[10] S. Savage, D. Wetherall, A. Karlin and T. Anderson, Practical Network Support for IP Traceback,, *Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, Stockholm, Sweden, 2000, 295-306.

[11] T. Peng C. Leckie and K. Ramamohanarao, Adjusted Probabilistic Packet Marking for IP Traceback, *Proceedings of Second International IFIP-TC6 Networking Conference*, Pisa, Italy, 2002, 697-708.

[11] T. Peng, C. Leckie and K. Ramamohanarao, Protection from Distributed Denial of Service Attack Using History-based IP Filtering, *Proceedings of the IEEE International Conference on Communications*, Anchorage, AK, 2003, 1-5.

[13] P. Ferguson and D. Senie, Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing, *RFC-2267, Internet Engineering Task Force*, 1998.

[14] L. Ricciulli, P. Lincoln, and P. Kakkar, TCP SYN Flooding Defense, *Communication Networks and Distributed Systems Modeling and Simulation*, San Diego, CA, 1999.

[15] X. Chen and J. Heidemann, Flash Crowd Mitigation via an Adaptive Admission Control Based on Application-Level Measurement, *Technical Report ISI-TR-557, University of Southern California/ Information Sciences Institute*, May 2002.

[16] J. Jung, B. Krishnamurthy, and M. Ravinovich, Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites, *Proceedings of the International World Wide Web Conference*, Honolulu, HI, 2002, 252-262.

[17] Jeffrey C. Mogul and K.K. Ramakrishnan, Eliminating Receive Livelock in an Interrupt-driven Kernel, *Proceedings of the USENIX Annual Technical Conference*, San Diego, CA, 1996, 99-111.

[18] Sun Microsystems, SUN's TCP SYN Flooding Solutions, URL: *http://ciac.llnl.gov/ciac/bulletins/h-02.shtml*, October 1999.

[19] A. Cox, Linux TCP Changes for Protection against the SYN attacks, URL: *http://www.wcug.wwu.edu/lists/netdev/199609/msg00091.html,* September 1996.

[20] S. Manley and M. Seltzer, Web Facts and Fantasy, *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, Anaheim, CA, 1997, 125-133.