Advanced Random Time Queue Blocking for Effective Protection of Application Servers Against Low-Rate DoS Attacks

R. Kavitha[1,2], G. Padmavathi[1]

(Corresponding author: R. Kavitha)

Avinashilingam Institute for Home Science and Higher Education for Women[1]

Department of Computer Science, Sri Krishna Arts and Science College[2]

Coimbatore, Tamilnadu, India

(Email: jayabal.kavitha@gmail.com)

(Received July 21, 2016; revised and accepted Nov. 15 & Dec. 25, 2016)

Abstract

Low-rate traffic denial-of-service (DoS) attacks are a strategy to deny services of a network by detecting the vulnerabilities in the application behaviors. The low-rate DoS attack against the application servers is considered in this paper with the motive to develop an efficient defense technique against the low-rate DoS attack. Among different defense techniques, the Improved Random Time Queue Blocking (IRTQB) performs better than other methods. IRTQB performs similar to Random Time Queue Blocking (RTQB), but it selectively chooses the blocking interval requests only from the potential attackers and discards them. However, the differentiation of the attacker requests from the legitimate users' is not always efficient as only the source IP addresses and the record timestamp are considered. This can be improved when considering more complex set of features. Hence, in this paper, the Advanced Random Time Queue Blocking (ARTQB) scheme is proposed by additionally employing Bandwidth utilization of attacker and legitimate user in IRTQB. ARTQB defines Spatial Similarity Metric (SSM) between the requests in terms of source IP addresses, the record timestamp and the bandwidth. Thus the defense of the application server against the low-rate DoS attack is be improved than IRTQB. Experimental results show that the proposed ARTQB performs better protection of Low-Rate DoS Attack against Application Servers (LoRDAS) by reducing the attack efficiency and attack impact on the server.

Keywords: ARTQB; IRTQB; Low-rate Denial-of-Service (LDoS); RTQB; Spatial Similarity Metric (SSM)

1 Introduction

Denial-of-service (DoS) is a type of attack in which the attackers attempt to prevent the legitimate users from accessing the network services. In a DoS attack, normally the attacker transmits unnecessary messages which are having invalid return addresses and requiring the network or server to authenticate requests [9]. While sending the authentication approval, the network or server has no ability to find the return address of the attacker, and causing the server to wait before closing the connection. The attacker transmits more authentication messages along with invalid return addresses while the server closes the connection. Hence, the process of authentication and server wait will begin again, keeping the network or server busy.

A network or host can be compromised with DDoS attacks using two types of traffic, namely, high-rate DoS traffic and low-rate DoS traffic [2]. DoS attacks are implemented in terms of many ways. The most common ways are the flooding the network to reduce the legitimate network traffic, disrupting the connections between the user and the server, blocking certain range of users and disrupting the state of the information of the users [10]. However the detection of DoS has become easier as it generates high inconsistent traffic rate by which the detection algorithms assures the presence of attack. Thus, low rate attacks came into real timer applications in which the DoS is achieved in low traffic scenarios.

Low-Rate DoS attacks (LRDoS) are new types of DoS attacks. In LRDoS the attacker sends a burst of well-timed packets, creating packet losses in a link and increments the retransmission timeout for only certain TCP flows. As these traffic bursts are sent during the expiration times, the overall traffic is reduced considerably thus disabling the efficiency of detection. Many techniques have been presented in the recent past to detect the LRDoS but most of the techniques performed below expectation. The introduction of new LRDoS such as Shrew and reduction of quality (RoQ) attacks increases the detection complexity.

In this paper, the DoS detection schemes such as random service time (RST), Random answer instant (RAI), Random time queue blocking (RTQB) and Improved Random Time Queue Blocking (IRTQB) are analyzed to determine the detection efficiency. The analysis results show that the IRTQB performs better than the other three methods; however the IRTQB also suffers from limitations. Particularly the differentiation between the attack requests from the legitimate users' requests is not satisfactory. Hence, bandwidth utilization is included in IRTQB to develop Advanced Random Time Queue Blocking (ARTQB) scheme for effective defense of the application servers.

The remainder of the article is organized as follows: Section 2 describes the related researches briefly. Section 3 presents the methodologies utilized in the paper. Section 4 provides the experimental results and their discussions. Section 5 concludes the research.

2 Related Works

Macia-Fernandez et al. [5] proposed evaluation method of low-rate DoS attack (LRDoS) against the iterative servers. The evaluated attack characteristics are analyzed and the potential effects of the attack are also analyzed. The iterative servers are that those servers limited to handle just a single service request compared to multiple service requests handled by the concurrent servers. The analysis provides the vulnerability details of the iterative servers and provides better possibility of forecasting the statistical metrics about the server behavior. The low-rate traffic behavior helps to subvert the provided service. It is also possible to tune the parameters of the attack in order to choose the suitable values for the efficiency and the amount of load generated in the target server. Thus, it becomes possible to bypass the intrusion detection system intended to protect the attacked server.

Macia-Fernandez et al. [6] in another work extended the analysis results to support the analysis of the low-rate DoS attacks against the concurrent servers like persistent HTTP servers. The mathematical model for the low-rate DoS attacks against the application server has been presented in an extended work [7] for the evaluation of the attack in servers with superposition among the occurrence probability functions. In another extension [8] presented four efficient methods to tackle the low-rate DoS attacks against the application server.

The efficient alternative techniques are based on the blocking the entry of the requests in the service queue of the server. Thus the efficiency of the low-rate DoS attack can be reduced effectively without any impact on the amount of time spent by the requests in the generated systems. However, there are some limitations in which the attack requests and the legitimate users' requests are not effectively differentiated in some instances.

Wang et al. [12] proposed a queuing analysis scheme for the evaluation of the DoS attacks in the computer networks. The stationary probability distribution can be determined by developing a memory-efficient algorithm and the computed probability distribution can be utilized for finding other interesting performance metrics like the connection loss probability and buffer occupancy percentages of half-open connections for regular traffic and attack traffic. Thus the impact of DoS attacks can be detected even in complicated computer networks.

Tang et al. [11] proposed a vulnerability model of feedback-control based internet services to tackle the low-rate DoS attacks. The fundamental queries, namely the impact of the LRDoS attack on the feedback-control based systems and how the systematic evaluation of LR-DoS is performed has been the center point of research. These problems are tackled by considering the target system as a switched system. Both the oscillation of steady state error and staying away from the desired state impair the system's performance and hence a novel methodology is used to analyze the impact of the attack. However the tradeoff between the effectiveness and the cost of LRDoS attack has not been explained which hinders the analysis.

Wu et al. [13] proposed an LRDoS attack detection scheme based on the network multi-fractal called as multi-fractal detrended fluctuation analysis (MF-DFA). The scheme detects the changes in terms of the multi-fractural characteristics of the network traffic, which helps in finding the LRDoS attack flows.

Adi et al. [1] introduced an analysis technique to demonstrate the impact of LRDoS attacks against the HTTP/2 servers. The resource consuming HTTP packets are transmitted along with the principle of sending requests in order to serve the full capacity of the servers. The HTTP/2 packets serve as the underlying standard and there are no computationally expensive applications due to a backend server, which is not connected to the HTTP/2 server. The server memory degrades at a certain rate indicating the presence of LRDoS.

Brynielsson et al. [4] presented a spectral analysis based detection of LRDoS attacks against the HTTP server. The weakness of the HTTP server is analyzed and the attack simulator has been developed. When the attack is present, disproportionate amounts of energy in the lower frequencies can be detected effectively. Thus the approach serves as a medium of detection with the attacker has fixed wait times or floods the server when initiating the attack. However the major drawback is that the attacker has certain approaches to reduce the disproportionate amounts of energy to some extend so that the attack detection becomes very difficult.

Bedi et al. [3] introduced the enhanced AQM technique called as Deterministic Fair Sharing (DFS) for tackling the congestion based DoS attacks. The concept of fair buffer share is dynamically determined for each competing flow to ensure optimal fairness. It is achieved in DFS by utilizing a set of data structures in combination to provide low operational overhead while maintaining limited per-flow state and offer high DoS attack identification capability. Thus the congestion based DoS attacks can be detected effectively and DFS provides a higher degree of fairness and throughput to legitimate flows while stabilizing the router queue length and allowing the least bandwidth to the attack traffic.

Though the methods discussed in the literature are effective in the defense against the low-rate DoS attacks, there seem to be more drawbacks that reduce the overall performance. In some methods, the LoRDAS attacks become complex to detect and hence the attack efficiency becomes not possible to be minimized. The tradeoff between the effectiveness and the cost of LRDoS attack influences the attack performance, but it is not considered in the existing methods. The smart attackers have the tendency of reducing the disproportionate amounts of energy in the existing methods also affects the attack detection.

Moreover, in the existing methods, the differentiation of the legitimate user requests and the attacker requests is appropriately addressed. Similarly, only the record timestamp and the source IP address are considered for reducing the LoRDAS attack efficiency, which seems efficient; however making room for improvement.

Hence the need for a novel strategy is needed to effectively detect the attacks and also develop a response technique for reducing the attack efficiency and its impact on the application server. Many methods were introduced to detect the attacks, including the Shrew attack.

Some methods utilized the randomization of the timers in the TCP flows for avoiding the synchronization between the periodic arrival of short attack bursts and the expiration of a timer. But there are no effective solutions for the LoRDAS. The Improved Random time queue blocking (IRTQB) has been the effective solution till date but even it has relatively near-par solutions only. IRTQB employs record timestamp and the source IP address for reducing attack impact. In this paper, ARTQB is proposed with the bandwidth utilization additionally considered for the minimization of the attack efficiency.

3 Methodologies

3.1 Application Server Model

Application servers are the potential victims of LoRDAS attacks shown in Figure 1. Certain conditions are required for an application to be vulnerable to this kind of attacks, and several different strategies might be followed by the attacker to deny the service. In addition, the necessary network model behind the attack is shown in Figure 2.

The application server model considered in the LoRDAS attack is composed of the following elements (1) a service queue where incoming requests are placed upon their arrival on the server, and (2) one or several service modules which are in charge of processing the requests.

The low-rate DoS attacks in the application servers depends on two major aspects of server behavior such as the presence of deterministic patterns and enabling instants concurrence with the answer instants. The defense methods are developed based on the strategy used for reducing

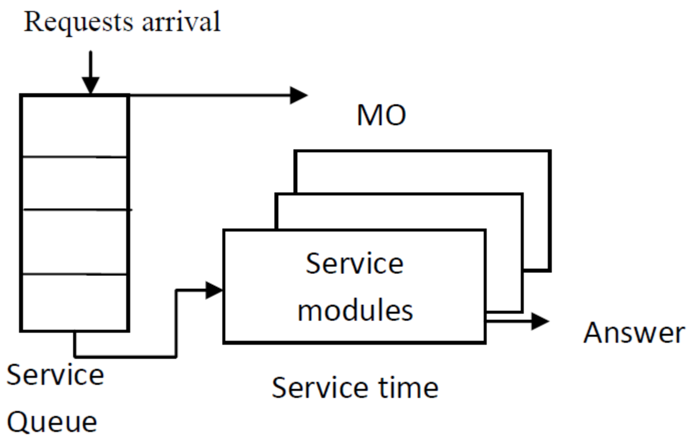

Figure 1: Application server model in LoRDAS attack

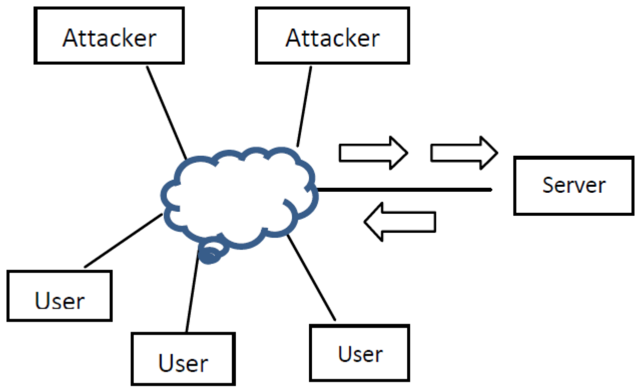

Figure 2: Study scenario

the efficiency of the attack in the servers and without any negative impact on the normal performance of the servers. The fundamental LoRDAS attack can be understood from Figure 3.

3.2 Random Service Time (RST)

RST has been designed with the view of reducing the predictability of the server behavior, thus considerably preventing the server details to the attackers and reducing the efficiency of the attack as shown in Figure 4. In this scheme, the deterministic patterns are eliminated from its mode of operation. So whenever the server utilizes a fixed timeout feature that aims to randomizing the features to make the implementation of the attack more difficult in practice by blocking the prediction of the answer instants and enabling instants. RST can be implemented in a server such that its behavior is maintained with slight modifications that do not alter the overall performance.

When the service of a request in the queue is idle, the service module considers the request from the service queue on the basis of the popular schemes like FIFO and LIFO. Then by utilizing a processing time of the request, the request is deadline constraint. In this period, the attacker manages to send a short attack burst that reaches around the estimated answer time. When the processing is finished, the service module remains locked for a random time called extra delay such that no additional position is enabled as no answers will be generated in this phase.

Either the conditions $\Delta t > \overline{\Delta t_{RST}} + B/2$ or $\Delta t < \overline{\Delta t_{RST}} - \frac{B}{2} - RTT$ must be achieved during the attack bursts so that the answer time is shifted. However condition $\Delta t < \overline{\Delta t_{RST}} - \frac{B}{2} - RTT$ cannot be fulfilled if $\overline{\Delta t_{RST}} < \frac{B}{2} + RTT$. Hence the Δt_{RST} takes values from a uniform distribution with a maximum value Δt^{RST}_{max} in order to maintain, no free positions for the legitimate users in RTT seconds.

$$\Delta t = U[0, \Delta t^{RST}_{max}], \text{ if} \overline{\Delta t_{RST}} < \frac{B}{2} + RTT \qquad (1)$$

where, $\overline{\Delta t_{RST}}$ is the mean extra delay, RTT is the round trip time, B is the time period of attack burst, Δt is the variability in service time and Δt^{RST}_{max} is the maximum value of mean extra delay. When $\overline{\Delta t_{RST} > \frac{B}{2} + RTT}$, Δt_{RST} is a random variable sampled from two different uniform variables V_1 and V_2 are utilized.

$$\overline{\Delta t} = \frac{\Delta t^{RST}_{max}}{2} \qquad (2)$$

The mean value of the extra delay should be $\frac{\Delta t^{RST}_{max}}{2}$ in order to appropriately shift the answer time. So Δt_{RST} is sampled from V_1 and V_2 with a probability P.

$$\Delta t_{RST} = \begin{cases} V_1 & \text{with probability } P \\ V_2 & \text{with probability } 1 - P \end{cases} \qquad (3)$$

where V_1 and V_2 are variables and the probability P is calculated by

$$P = \frac{\Delta t^{RST}_{Max} + B}{2(\Delta t^{RST}_{Max} + B + RTT)} \qquad (4)$$

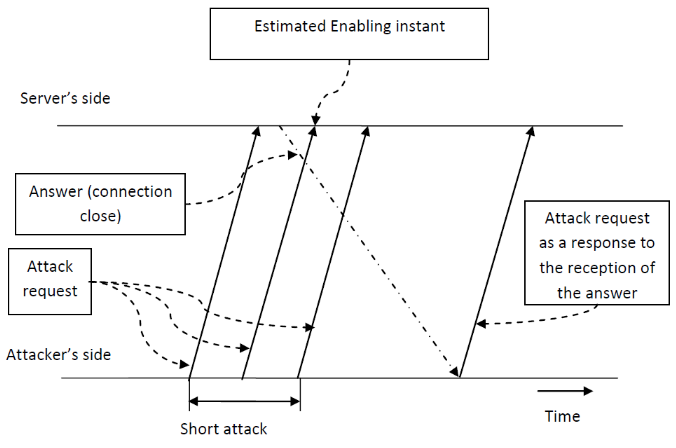
Estimated Enabling instant
Server's side
Answer (connection close)
Attack request
Attack request as a response to the reception of the answer
Attacker's side
Short attack
Time

Figure 3: LoRDAS attack process

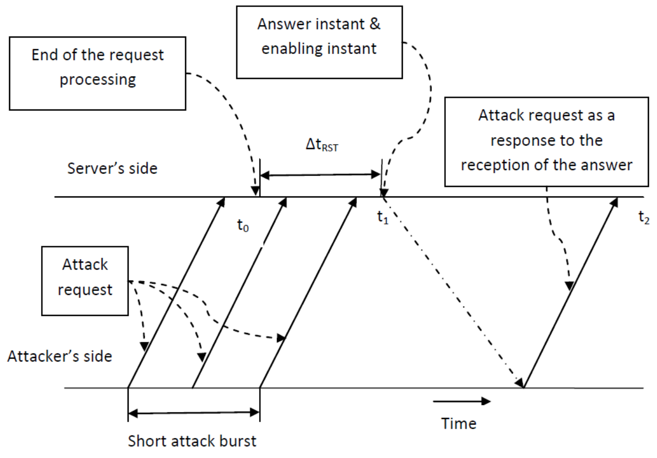
End of the request processing
Answer instant & enabling instant
Attack request as a response to the reception of the answer
Server's side
Δt_RST
t0
t1
t2
Attack request
Attacker's side
Short attack burst
Time

Figure 4: LoRDAS attack when RST is active

The RST initially replaces the answer instant to a newer position which does not come under the control of the attacker. If the extra delay is longer and the condition of the service queue is full of requests, then all of the attack burst traffic will be rejected by the server. The time available for the legitimate user to insert new requests in the service queue depends on the round trip time (RTT) between the server and the attacker while the reception of the attack packet will be sent as a response to the answer.

When the extra delay time expires, the answer is forwarded to the user who requested them. As the efficiency of the attack is reduced, the server performs efficiently and thus a new free position occurs at the end of transmitting the answers to the requested users. If the extra delay time is much longer, then the legitimate user can send a request at the start of the delay time while the attacker's request is assumed to be occurring at the end of the delay time, so that even the attack requests can be inserted at the free positions in the queue without affecting the user requests.

Thus the efficiency of the attack is significantly reduced; however the fact assumed that the attack requests are inserted at the end of the lengthy extra delay. But when an extra delay is added to the original service time, the attacker also perceives an increase in the estimation of the service time, so that the attack parameters can be adjusted to synchronize the attack bursts.

3.3 Random Answer Instant (RAI)

Random answer instant (RAI) differs from the RST technique by utilizing the decoupling the answer instants and the enabling instants instead of introducing variability in the server behavior as performed in RST as shown in Figure 5. The service of a request is extracted and processed during the service time as in the RST. After processing the requests from the queue, the service model waits for the extra delay similar to RST but the difference being when the first request service is processed, and then the new request is extracted from the queue and begins processing such that the extra delay becomes non-blocking. Thus a new position in the queue is enabled at this period, enabling instant. After the completion of extra delay, the answer is sent to the corresponding user which is the answer instant.

When $\overline{T_S} + \overline{\Delta t_{RAI}}$ is the service time, a short attack burst is send at the answer instant $t_1 - t_0$. Δt is given by $\Delta t = U[\Delta t^{RAI}_{min}, \Delta t^{RAI}_{max}]$. The lower limit is given as $\Delta t^{RAI}_{min} = \frac{B}{2}$ thus ensuring the time interval duration $\overline{\Delta t_{RAI}} - B/2$ between the enabling instant and the arrival of the short attack burst.

$$\Delta t = \begin{cases} 0 & \text{if } \Delta t^{RAI}_{max} < B/2 \\ U[\Delta t^{RAI}_{min}, \Delta t^{RAI}_{max}] & \text{if } \Delta t^{RAI}_{max} > B/2 \end{cases} \quad (5)$$

where, Δt^{RAI}_{max} is the upper limit of the uniform distribution to select Δt_{RAI} and Δt^{RAI}_{min} is the lower limit of the uniform distribution to select Δt_{RAI}. The upper limit Δt^{RAI}_{max} must be high as possible in order to introduce higher variability. But by considering impact the values becomes

$$t^{RAI}_1 = [Nt_s + \Delta t_{RAI}, (N+1)t_s + \Delta t_{RAI}] \quad (6)$$

In Equation (6), t^{RAI}_1 is the time interval for new incoming request in RAI mechanism, t_s is the service time, Δt_{RAI} is the mean extra delay in RAI, N is the number of requests. Thus Δt^{RAI}_{max} should be configured as a trade-off between reducing the impact and increasing the variability in the server so that RAI reduces the impact on the normal behavior of the server so that the effectiveness of the attack is reduced. However there is an interval called as the tradeoff between the reduction of the impact and increase of variability in the server, which causes impact on the normal behavior of the server.

3.4 Random Time Queue Blocking (RTQB)

As RST and RAI have certain limitations, Random time queues blocking (RTQB) is introduced to overcome the shortcomings as shown in Figure 6. RTQB aims at reducing the attack efficiency without creating any negative impact in the server behavior. The main concept of RTQB is that when the attacker is able to accurately estimate the answer instants, then the short attack bursts will arrive with the response attack messages arriving in RTT seconds. In this situation, the legitimate users are distributing the requests while the attackers will be considering the response messages. Hence, in RTQB all the requests arriving at this time interval are blocked so that the attack efficiency and the impact are reduced.

Ideally, the value for $\Delta t_{RTQB} = RTT$, all the attack requests are projected in an interval $[-B/2, RTT]$ around the answer instant. But Δt_{RTQB} is configured as a random value taken from uniform distribution

$$\Delta t_{RTQB} = U[RTT, \Delta t^{RTQB}_{max}] \quad (7)$$

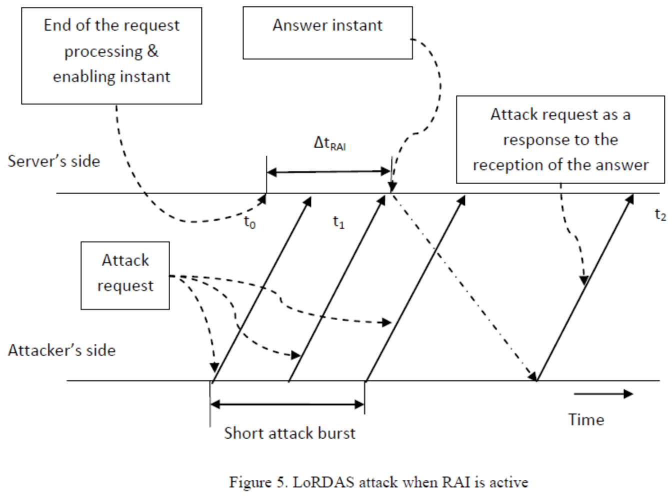
End of the request processing & enabling instant
Answer instant
Attack request as a response to the reception of the answer
Server's side
Δt_{RAI}
t_0
t_1
t_2
Attack request
Attacker's side
Short attack burst
Time
Figure 5. LoRDAS attack when RAI is active

Figure 5: LoRDAS attack when RAI is active

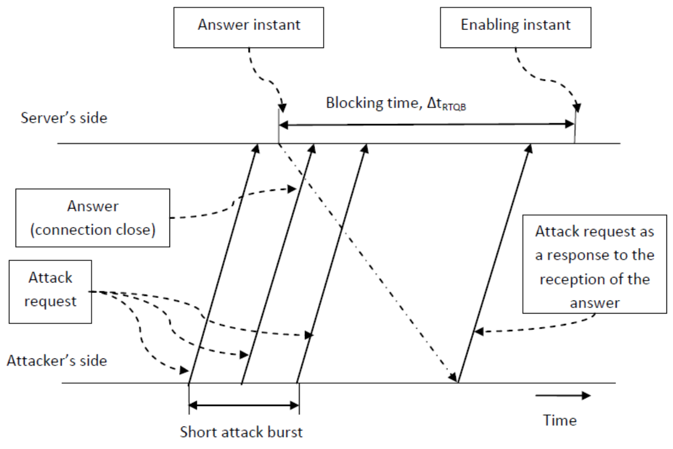
Answer instant
Enabling instant
Blocking time, Δt_{RTQB}
Server's side
Answer (connection close)
Attack request
Attack request as a response to the reception of the answer
Attacker's side
Short attack burst
Time

Figure 6: LoRDAS attack when RTQB is active

In Equation (7), Δt_{RTQB} is the mean extra delay for RTQB. There are two reasons for considering a random value. First, the attacker estimation of both the answer instants and RTT is not perfect and, attack packets will arrive even after RTT seconds from the answer instant. Second, it is recommendable to introduce certain variability into the process; the attacker might be capable of estimating the value Δt_{RTQB} and adapting the attack.

The service of a request is performed similar to the RST and RAI. After the processing of the requests, the answers are sent to the requested users in answer instant and the extraction of new requests is enabled from the queue in enabling instant. After the answer instant, all new requests are rejected during the specified interval of time. After this interval, the new requests are again accepted and the processing begins. Thus the effect of an attack can be reduced while once a request enters during the active stage of RTQB; the server behavior is also not affected. The impact of RTQB on the server behavior, it is clear that when RTQB is active and once a request enters into the service queue, there is no difference in its process. Thus, no impact is present due to the use of RTQB. However, as RTQB blocks all the requests during the specified time interval, it fails to select the legitimate user requests.

3.5 Improved Random Time Queue Blocking (IRTQB)

In order to avoid discarding of important legitimate user requests during the time interval of RTQB, improved version called Improved Random Time Queue Blocking (IRTQB) is introduced as shown in Figure 7. IRTQB selectively chooses the requests during the time interval around the answer instants which comes from the potential attackers are only discarded. As the attacker uses only restricted spoofing mechanisms that are allowed within the same network segment, the attack is limited. The attack interval starts only at $t_0 - B/2$ where B is the length of an attack burst. Thus, when the attack length is B, the interval becomes $[t_0 - B/2, t_0 + \Delta t_{max}^{IRTQB}]$. The spatial similarity metric SSM is defined between the two requests to measure the probability that they come from the same source. In IRTQB, the SSM includes the source IP addresses only. For two generic IP addresses A_i and A_j, the similarity metric is computed as the number of consecutive bits set to '1' in the bit XNOR operation of the two addresses.

$$SSM(A_i, A_j) = \#\_consecutive\_bits_1(A_i \text{ XNOR } A_j) \quad (8)$$

A spatial similarity metric (SSM) is defined between two requests to measure the probability of the same source. SSM includes the source IP addresses of incoming requests for the analysis of the source. IRTQB maintains a record of timestamps and source IP addresses for all incoming requests with which the similarity is computed. If the similarity between the two requests is higher than a predefined threshold $SSM(A_i, A_j) > SSM_{Th}$ then, both are discarded without any notification to the users. Thus, the attackers are prevented from obtaining information about the requests.

3.6 Limitations of RST, RAI, RTQB, IRTQB

RST decreases the attack efficiency by shifting the answer time to a position that is not controlled by the attacker and also by adding a source of variability in the server behavior. However, when the extra delay included in the original service time is longer, the attacker also perceives an increase in the estimation of the service time, so that the attack parameters can be adjusted to synchronize the attack bursts. The reduction of attack effectiveness in RST is limited due to the fact that the maximum amount of time for legitimate users to seize new positions in the queue is the RTT. RAI technique performs better than RST and even provides better performance than the RTQB. However, in RAI the tradeoff between the reduction of the impact and increase of variability on the server, which causes an impact on the normal behavior of the server. As the defense technique should not cause any impact on the server, RAI is avoided and RTQB is presented. RTQB reduces the impact on server while also reducing the attack efficiency. It makes advantage of the attackers using short bursts of traffic that arrive around the answer instants and blocks all the incoming requests in a time interval. During this interval RTQB does not selectively choose the requests and blocks all requests without analyzing the request sender. This becomes a major drawback in RTQB which leads to an improved technique called IRTQB. IRTQB employs SSM and selectively blocks the requests so that the queue has at least one free space and the attack efficiency is reduced.

Though IRTQB performs better than RST, RAI and RTQB by reducing attack efficiency and no impact on the server, the SSM metric only includes the source IP address and record timestamps. By including more reliable factors, the performance can be further improved. Thus the need for more advanced defense technique with SSM considering more factors arises. This need is achieved by including the bandwidth utilization factor with source IP address and record timestamps in the proposed ARTQB defense technique.

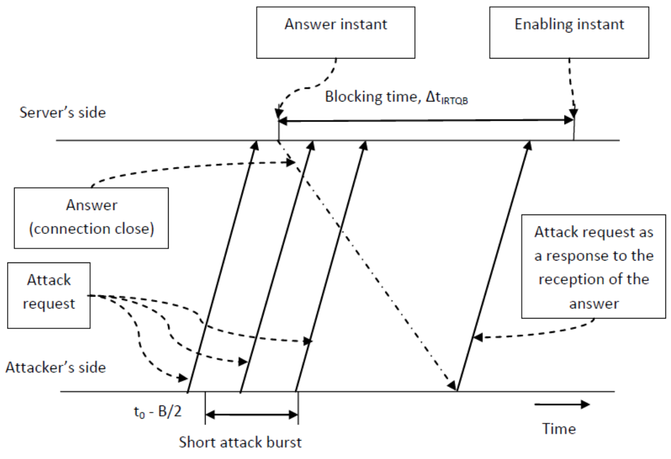

Figure 7: LoRDAS attack when IRTQB is active

3.7 Advanced Random Time Queue Blocking (ARTQB)

Though IRTQB significantly reduces the attack efficiency without impact on server behavior, still there is scope for improvement in reducing the attack efficiency. IRTQB is very efficient, but at the same time it is more costly than other methods and hence improving the performance without further increasing the cost is much more significant. Hence, a highly improved version of RTQB is proposed which is called as Advanced Random Time Queue Blocking (ARTQB) as shown in Figure 8. Consider the utilization bandwidth of user b_u, is employed by the service in processing a given request. The service bandwidth for legitimate request and attack requests are varied for their processing. Therefore, the service bandwidth for identical requests is modeled as a random variable B_u, with the normal distribution. The mean value of B_u is denoted as $\overline{B_u}$ and the variance is denoted as $var[B_u]$.

$$B_u = N(\overline{B_u}, var[B_u]) \tag{9}$$

Where, N refers the normal distribution. Consider Δb_{ARTQB} is the extra bandwidth utilized and the value for $\Delta b_{ARTQB} = var[B_u]$, as all the attack requests are expected in an interval $[\overline{B_u}, max(var[B_u])]$ around the answer instant. Hence, Δb_{ARTQB} is configured as a random value which is taken from the uniform distribution:

$$\Delta b_{ARTQB} = U[\overline{B_u}, \Delta b_{max}^{ARTQB}] \tag{10}$$

Where, Δb_{max}^{ARTQB} is the maximum value of Δb_{ARTQB} and U refers the uniform distribution. Assume, the bandwidth utilization of the attacker's requests are around the answer instants are very high. Assume the queue contains $N-1$ requests and that they are all processed by the service module at the rate of bandwidth b_u Hz per request. The attack bandwidth b_1 for new incoming request is computed as,

$$b_1^{ARTQB} = [Nb_u + \Delta b_{ARTQB}, (N+1)b_u + \Delta b_{ARTQB}] \tag{11}$$

Therefore, the similarity metric SSM, is measured between two requests in order to identify the probability that they come from the same source including same bandwidth. Similar to Equation (8), the spatial similarity metric is computed as the number of consecutive bits set to '1' in the bit XNOR operation of the two bandwidths by considering two bandwidths B_i and B_j:

$$SSM(B_i, B_j) = \#\_consecutive\_bits_1(B_i \text{ XNOR } B_j) \tag{12}$$

Then, the simple spatial similarity metric for both bandwidth and IP addresses is computed as,

$$\begin{aligned} & SSM(A_i, A_j, B_i, B_j) \\ = \ & SSM(A_i, A_j) + SSM(B_i, B_j) \qquad (13) \\ = \ & \#\_consecutive\_bits_1(A_i \text{ XNOR } A_j) \\ & \quad + \#\_consecutive\_bits_1(B_i \text{ XNOR } B_j) \end{aligned} \tag{14}$$

3.8 Description

ARTQB extracts the requests from the service queue and processes them at the service time T_s. After completing the processing the answers are send to the legitimate users who requested them while on the other side called enabling instant, the new requests are started to process. These requests arriving at attack interval t_1 and attack

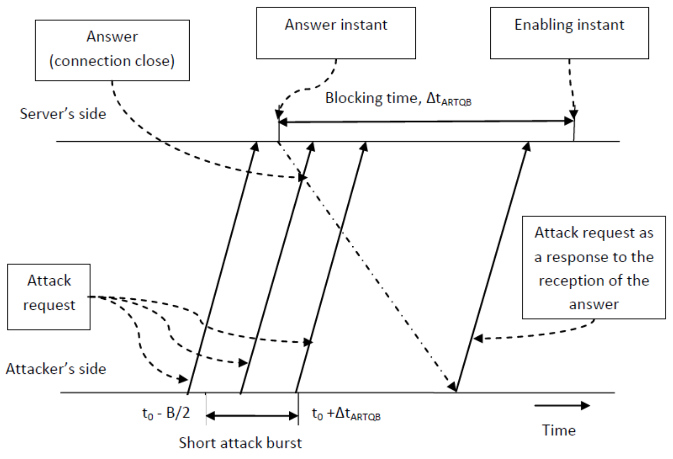

Figure 8: LoRDAS attack when ARTQB is active

Algorithm 1 ARTQB execution

```
1: Extract request from service queue
2: Processing request at service time $T_s$
3: //Answer generation
4: For every answer
5: Insert answer instant in a list L
6: Compute attack interval $t_1$
7: Compute attack bandwidth $b_1$
8: Send answer to users
9: Extract new request (enabling instant)
10: End For
11: For every incoming request $R_a$
12: Record timestamp, source IP, bandwidth utilization
13: Determine $t_1$ and $b_1$ for $R_a$
14: For all requests $R_b$ in $t_1$ and $b_1$
15: Compute SSM
16: Determine a threshold for SSM, $SSM_{Th}$
17: if $(SSM(R_a, R_b) > SSM_{Th})$ then
18:     Discard $R_a$ and $R_b$
19: else
20:     Insert Request in Queue
21: end if
22: End For
```

bandwidth b_1 are selectively chosen and those from the attackers are discarded. When the attack interval expires, new requests are accepted again and the processing begins. ARTQB maintains a list of answer instants from the beginning of the server operation. Similar to IRTQB, around every answer instant, the attack interval and attack bandwidth exists indicating that the attack packets will be arriving during the interval and bandwidth. This helps in avoiding those attacks in the form of requests. The attack interval starts at the length of attack burst B which means that the interval begins at halfway through the initial time and the attack bandwidth starts that the bandwidth begins at halfway through the initial bandwidth. At this instant, no requests will enter the queue but these will be helpful in deciding the new requests as legitimate or attack requests.

ARTQB also maintains a list containing the timestamps, source IP addresses and bandwidth utilization for all incoming requests. Using the record list, for all the incoming requests the SSM is computed between the incoming request R_a and every request R_b arriving in the attack interval and bandwidth. If the SSM is higher than the defined threshold SSM_{Th}, both the requests R_a and R_b are discarded while in other case the incoming request is accepted. Thus the attack efficiency is reduced below half when the bandwidth utilization is included along with timestamps and source IP addresses in the SSM metric. It is also noted that there is no impact in the server behavior when the ARTQB scheme is active in the server. The overall flow of ARTQB is shown in Figure 9.

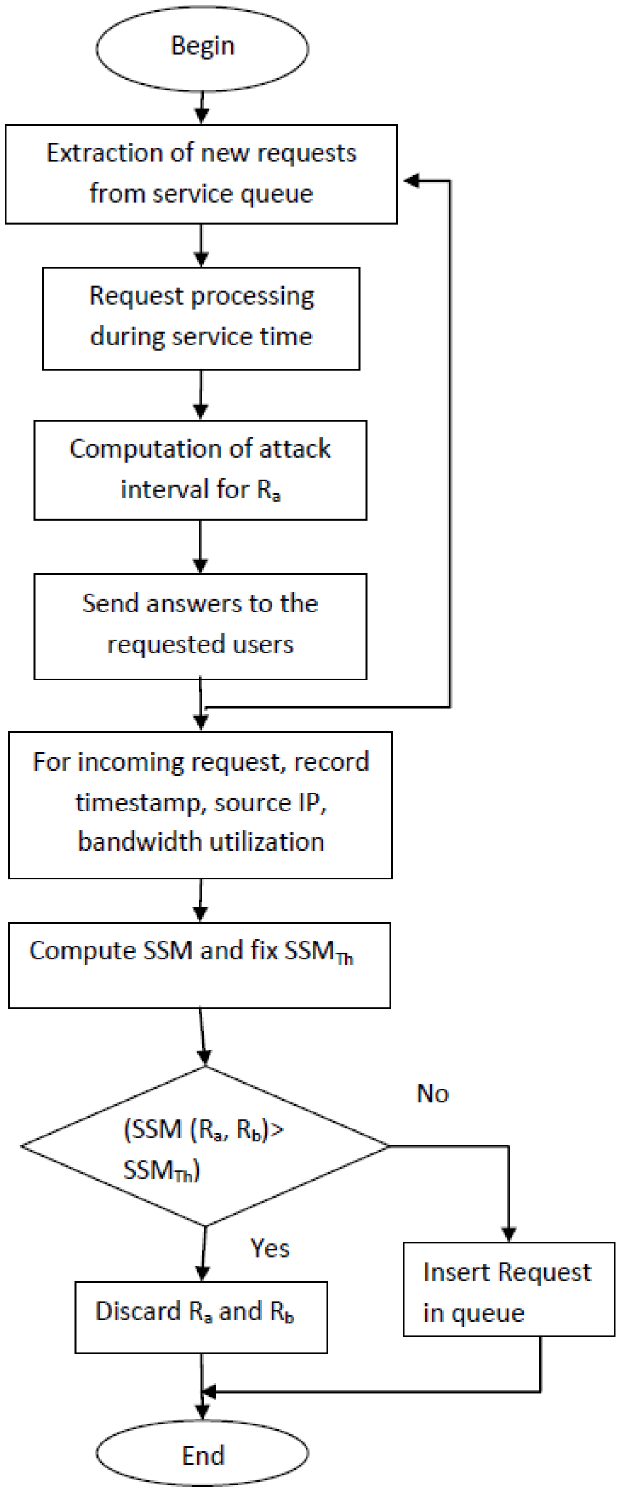
Begin
Extraction of new requests from service queue
Request processing during service time
Computation of attack interval for Ra
Send answers to the requested users
For incoming request, record timestamp, source IP, bandwidth utilization
Compute SSM and fix SSMTh
(SSM (Ra, Rb)> SSMTh)
No
Yes
Discard Ra and Rb
Insert Request in queue
End

Figure 9: Overall flow of ARTQB

4 Performance Evaluations

In this section, the proposed defense techniques are evaluated experimentally in the Network Simulator-2. The performance of the low-rate DoS attack is evaluated by measuring the mean in-system time and the attack efficiency. The low-rate DoS attack is employed in the application server inorder to evaluate the attack impact in the server. The server is supplied with different defense techniques namely RST, RAI, RTQB, IRTQB and ARTQB with configuration parameters prescribed with IRTQB and ARTQB considering additional configuration parameter called the SSM threshold. Table 1 shows the configuration values for attack and server parameters in Scenario 1 (S_1), Scenario 2 (S_2) and Scenario 3 (S_3).

Table 1: Configuration values for the attack and server parameters

Parameter	Value
Duration of attack burst, B	0.4s
Time between attack packets in a burst	0.2s
Mean service time, Ts	12s
Variance of server, $var[T_s]$	$0(S_1), 0.2(S_2, S_3)$
Interval between legitimate users requests	$3s(S_1, S_2), 0.95s(S_3)$
Number of server threads	$1(S_1, S_2), 4(S_3)$
Number of positions in service queue, N	$4(S_1, S_2), 8(S_3)$
Number of attack threads	-N
Round trip time, RTT	1s
Similarity metric, ST	32

Scenario 1, S_1: The server is mono-threaded and the variance of the service time for attack requests, $var[T_s]$, and $var[B_s]$ is 0. Scenario 2, S_2: Server variance of $var[T_s]$, and $var[B_s]$ is modified. Scenario 3, S_3: The aim of this scenario is to check how a multithread operation in the server affects the performance of a given defense technique. The Figures 10, 11, and 12 show the comparison of RST, RAI, RTQB, IRTQB and ARTQB in terms of attack efficiency (%) in three scenarios S_1, S_2 and S_3. The Figures 13, 14, and 15 show the comparison in terms of mean in-system time (s) in scenarios S_1, S_2 and S_3.

4.1 Attack Efficiency

Attack efficiency is the percentage of service queue positions captured by the attacker over the total number of positions captured during the attack execution.

Figure 10 shows that the attack efficiency comparison of RST, RAI, RTQB, IRTQB and ARTQB during Scenario 1 (mono-threaded with zero variance). In Scenario 1, the attack efficiency decreases from 100% to an asymptotic value. The attack efficiency of ARTQB has been much reduced than the other mechanisms since, consideration of the bandwidth utilization. It shows that the 100% attack efficiency in RST decreases to 39% for ARTQB whereas the attack efficiency of other mechanisms such as RAI, RTQB and IRTQB are 90%, 82% and 60% in the time period of 2sec. When the time period is 12sec, the attack efficiency of ARTQB is 20% which is lower than the other defense mechanisms.

Figure 11 shows that the attack efficiency comparison of RST, RAI, RTQB, IRTQB and ARTQB during Scenario 2 (di- threaded with variance 0.2). In Scenario 2, the attack efficiency decreases from 85% to an asymptotic value. By considering the bandwidth utilization in ARTQB, the attack efficiency is reduced compared with other mechanisms. It shows that when the time period is 2sec, the attack efficiency of ARTQB is 54% which is smaller than the other mechanisms. Also, it is clear that the 85% attack efficiency in RST decreases to 27% for ARTQB whereas the attack efficiency of other mechanisms such as RAI, RTQB and IRTQB are 51%, 41% and 35% in the time period of 12sec.

Figure 12 shows the attack efficiency comparison of RST, RAI, RTQB, IRTQB and ARTQB during Scenario 3 (multi-threaded with variance 0.2). In Scenario 3, the attack efficiency of ARTQB is much reduced than the IRTQB technique by considering the bandwidth utilization. It shows that when the time period is 2sec, the attack efficiency value of ARTQB is 46% compared with other mechanisms. Moreover, it is clear that the 87% attack efficiency in RST decreases to 23% for ARTQB whereas the attack efficiency of other mechanisms such as RAI, RTQB and IRTQB are 55%, 44% and 35% in the time period of 12sec.

4.2 Mean In-system Time

Mean in-system time is the time from when a request enters the server to the instant at which its corresponding answer is sent.

Figure 13 shows that the mean in-system time comparison of RST, RAI, RTQB, IRTQB and ARTQB during Scenario 1 (mono-threaded with zero variance). In Scenario 1, the mean in-system time decreases from RST to ARTQB. The reduction in mean in-system time is achieved by considering the bandwidth utilization. It shows that the 87sec in RST decreases to 30sec for ARTQB whereas the mean in-system time of other mechanisms such as RAI, RTQB and IRTQB are 71sec, 49sec and 32sec in the time period of 2sec. When the time period is 12sec, the mean in-system time of ARTQB is 22sec which is lower than the other defense mechanisms.

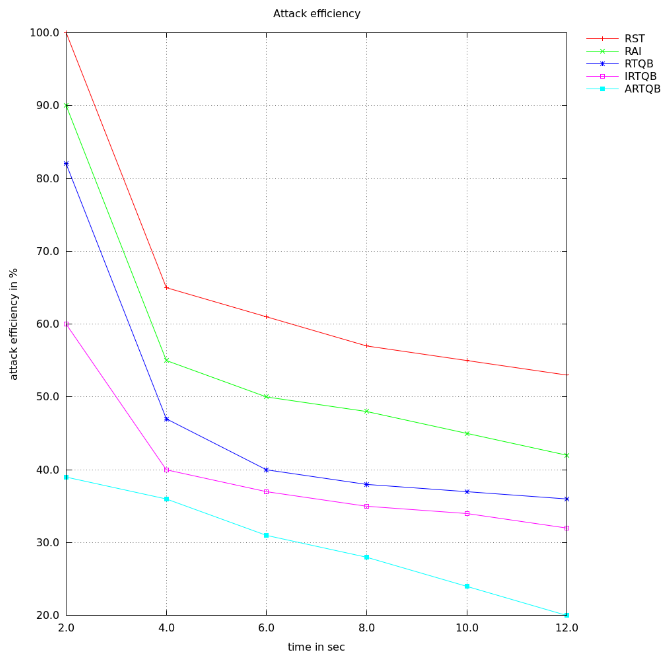

Figure 10: Scenario 1 attack efficiency (%)

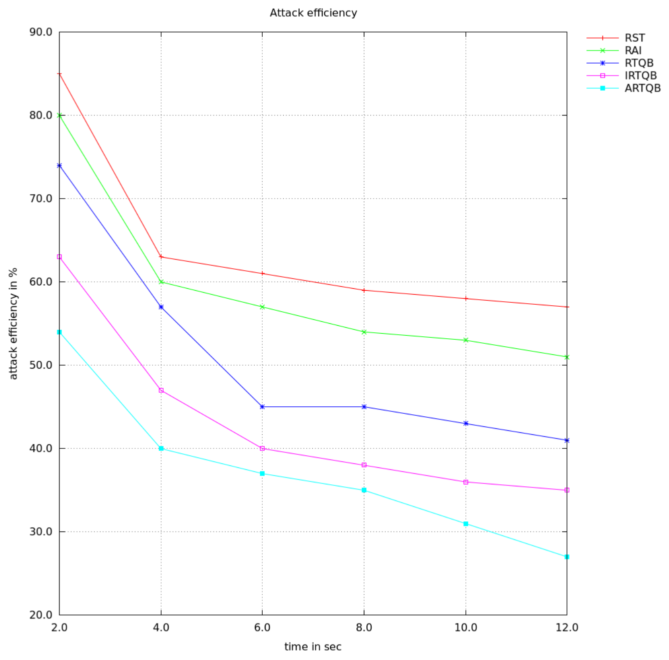

Figure 11: Scenario 2 attack efficiency (%)

Figure 14 shows that the mean in-system time comparison of RST, RAI, RTQB, IRTQB and ARTQB during Scenario 2 (di-threaded with variance 0.2). In Scenario 2, the mean in-system time decreases from RST to ARTQB.

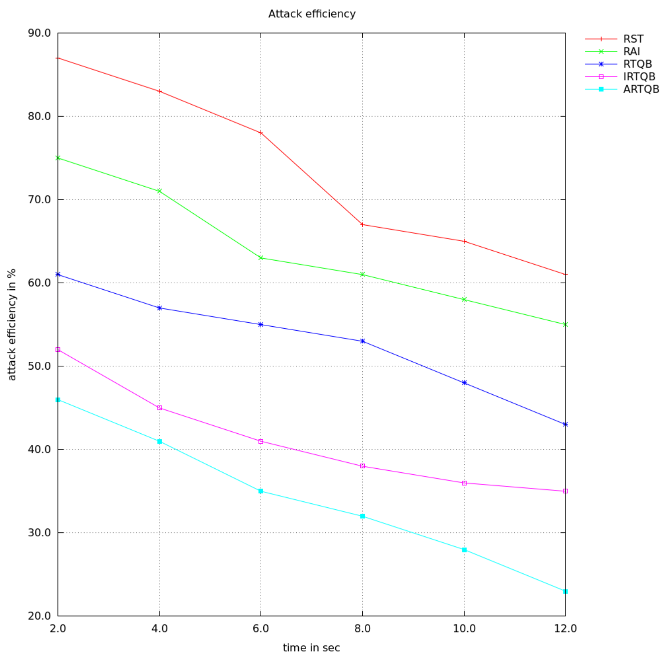

Figure 12: Scenario 3 attack efficiency (%)

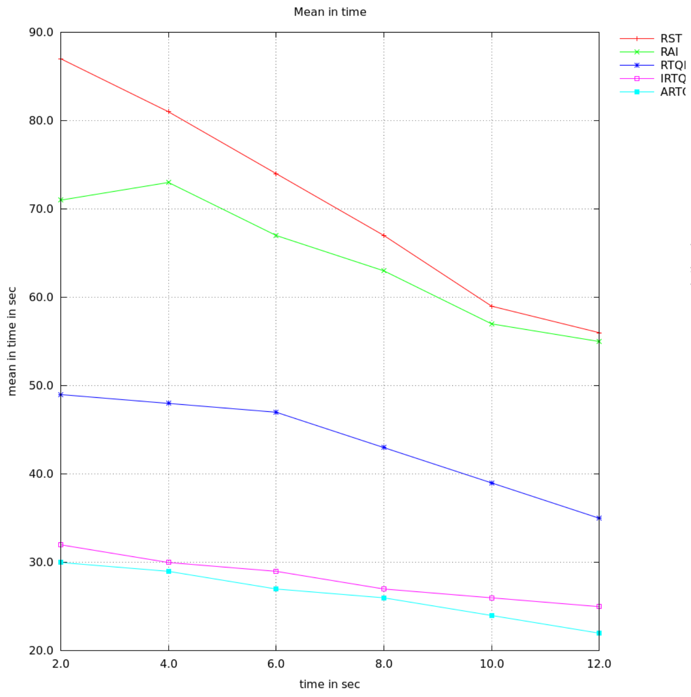

Figure 13: Scenario 1 mean in-system time (s)

The mean in-system time of ARTQB has been decreased due to considering the bandwidth utilization. It shows that when time period is 2sec, the mean in-system time of ARTQB is 41sec compared with other defense mechanisms. Moreover, it is clear that the 56sec mean in-system time in RST decreases to 25sec for ARTQB whereas the mean in-system time of other mechanisms such as RAI, RTQB and IRTQB are 55sec, 35sec and 29sec in the time period of 12sec.

Figure 15 shows that the mean in-system time comparison of RST, RAI, RTQB, IRTQB and ARTQB during Scenario 3 (multi-threaded with variance 0.2). In Scenario 3, the mean in-system time of ARTQB is much reduced than the RST technique by considering the bandwidth utilization. It shows that when the time period is 2sec, the mean in-system time of ARTQB is 40sec compared to the other mechanisms. Also, it is clear that the 38sec mean in-system time of RST decreases to 23sec for ARTQB whereas the mean in-system time of other mechanisms such as RAI, RTQB and IRTQB are 34.1sec, 28sec and 27sec in the time period of 12sec.

From the overall results the major research outcomes are that the RST is very simple defense yet not the best method as the attack efficiency is high and also there is impact of attack in the server behavior. When the extra delay is smaller, RTQB and IRTQB performs better but for higher extra delay RAI outperforms the other methods. However RAI also does not reduce the attack impact on server especially during the extra delay.

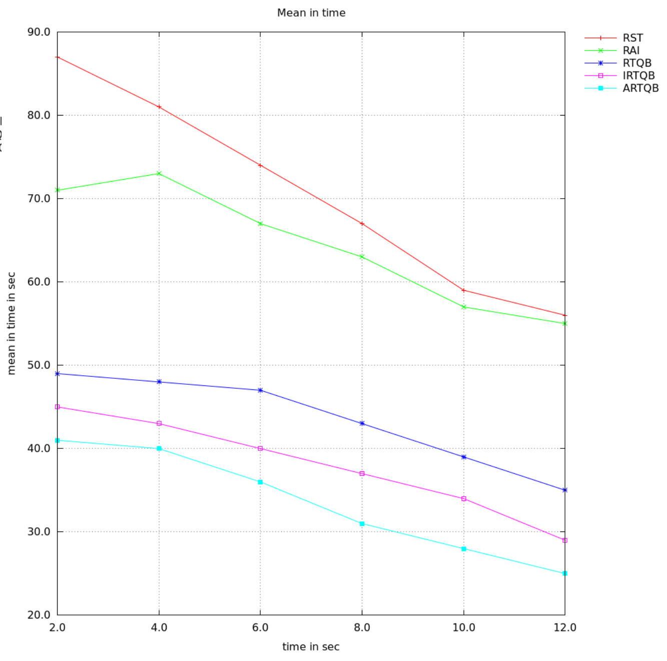

Figure 14: Scenario 2 mean in-system time (s)

Even RTQB and IRTQB have limitations. Considering these aspects the proposed ARTQB uses SSM with bandwidth utilization reducing the attack efficiency and the impact on the server more than half of the initial efficiency without further increasing the cost.

5 Conclusion

Detection of the low-rate DoS attacks is very important in ensuring the behavior of the application servers. Though

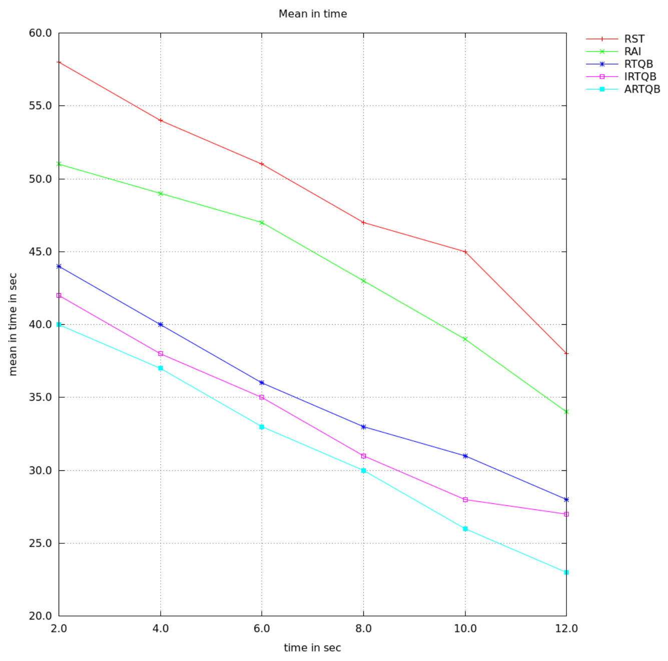

Figure 15: Scenario 3 mean in-system time (s)

the methods such as RST, RAI, RTQB and IRTQB reduces the effect of low-rate DoS attack without much impact on the server behavior, still there are limitations. Hence in this paper, ARTQB is proposed with the aim of maximal reduction of the attack efficiency on the server and minimizing the impact on server behavior. ARTQB selectively chooses the requests during answer instants. Similarly the use of SSM with the bandwidth utilization along with considering source IP addresses and the record timestamp enhances the reduction of attack efficiency. Experimental results conclude that the proposed ARTQB reduces the attack efficiency below half without any impact on the application server behavior.

References

[1] E. Adi, Z. Baig, C. P. Lam, and P. Hingston, "Low-rate denial-of-service attacks against HTTP/2 services," in *5th IEEE International Conference on IT Convergence and Security (ICITCS'15)*, pp. 1–5, 2015.

[2] A. Ain, M. H. Bhuyan, D. K. Bhattacharyya and J. K. Kalita, "Rank correlation for low-rate DDoS attack detection: An empirical evaluation," *International Journal of Network Security*, vol. 18, no. 3, pp. 474–480, 2016.

[3] H. Bedi, S. Roy, and S. Shiva, "Mitigating congestion based DoS attacks with an enhanced AQM technique," *Computer Communications*, vol. 56, pp. 60–73, 2015.

[4] J. Brynielsson, and R. Sharma, "Detectability of low-rate HTTP server DoS attacks using spectral analysis," in *Proceedings of IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pp. 954–961, 2015.

[5] 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.

[6] G. Maciá-Fernández, J. E. Díaz-Verdejo, and P. García-Teodoro, "Evaluation of a low-rate DoS attack against application servers," *Computers & Security*, vol. 27, no. 7, pp. 335–354, 2008.

[7] G. Maciá-Fernández, J. E. Díaz-Verdejo, and P. García-Teodoro, "Mathematical model for low-rate DoS attacks against application servers," *IEEE Transactions on Information Forensics and Security*, vol. 4, no. 3, pp. 519–529, 2009.

[8] G. Maciá-Fernández, Rafael A. Rodr?guez-G?mez, and J. E. Díaz-Verdejo, "Defense techniques for low-rate DoS attacks against application servers," *Computer Networks*, vol. 54, no. 15, pp. 2711–2727, 2010.

[9] J. Mirkovic, and P. Reiher, "A taxonomy of DDoS attack and DDoS defense mechanisms," *ACM SIGCOMM Computer Communication Review*, vol. 34, no. 2, pp. 39–53, 2004.

[10] A. Shevtekar, J. Stille, and N. Ansari, "On the impacts of low rate DoS attacks on VoIP traffic," *Security and Communication Networks*, vol. 1, no. 1, pp. 45–56, 2008.

[11] Y. Tang, X. Luo, H. Qing, and R. K. Chang, "Modeling the vulnerability of feedback-control based internet services to low-rate DoS attacks," *IEEE Transactions on Information Forensics and Security*, vol. 9, no. 3, pp. 339–353, 2014.

[12] Y. Wang, C. Lin, Q. L. Li, and Y. Fang, "A queueing analysis for the denial of service (DoS) attacks in computer networks," *Computer Networks*, vol. 51, no. 12, pp. 3564–3573, 2007.

[13] Z. J. Wu, L. Zhang, and M. Yue, "Low-rate DoS attacks detection based on network multifractal," *IEEE Transactions on Dependable and Secure Computing*, vol. 13, no. 5, pp. 559–567, 2016.

Biography

R. Kavitha is a Ph.D. research scholar of Avinashilingam Institute for Home Science and Higher Education for women, currently doing research on cyber security. She has 10 years of teaching experience in Sri Krishna Arts and Science College; Coimbatore. Her areas of interest include Low Rate Denial of Service Attack

Dr. G. Padmavathi is the Professor and Head of computer science of Avinashilingam Institute for Home Science and Higher Education for women, Coimbatore. She has 23 years of teaching experience and one year of industrial experience. Her areas of interest include Real Time Communication, Network Security and Cryptography. She has 200 publications in her research area.

Presently she is guiding M.phil researcher and PhD's Scholar. She has been profiled in various Organizations her academic contributions. She is currently the principal investigator of four projects funded by UGC and DRDO. She is the scientific mentor for one project funded by DST. She is life member of many preferred organizations of CSI, ISTE, WSEAS, AACE, and ACRS.