DDOS Mitigation In Cloud Computing Environment By Dynamic Resource Scaling With Elastic Load Balancing

A. Somasundaram[a], Dr. V. S. Meenakshi [b]

[a]Research Scholar, PG & Research Department of Computer Science, Chikkanna Government Arts College, Tirupur,
[b]Assistant Professor, PG & Research Department of Computer Science, Chikkanna Government Arts College, Tirupur,
[a]somasundaram.a@gmail.com, [b]meenasri70@yahoo.com

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 10 May 2021

Abstract: The major attack in cloud computing is Distributed Denial of Service (DDoS) which brings more attention to cloud users in the past decade. A DDoS attack can be avoided or controlled by allocating enough resources whenever demanded. Resource allocation will induce additional costs when the attacks reside for a long duration or more frequently. To control and avoid the DDoS attacks dynamic resource allocation has been employed in specific target services for mitigating the attack effectively. During the attack, the resources are overloaded with higher usage and lead to denial of required service for legitimate users forcefully. By adjusting the resource utilization, the attacks can be completely mitigated by which the genuine users' request for the resources can be apparently handledbesidesthe disbandment of established connection with the attacker's node. Thus resource utilization factor plays a significant role in attack mitigation and recovery which is the number of various resources allocated to the victim service. In this research, a new method has been proposed to evaluate the resource utilization by 'scaling down the resources'which is an enhancement of the 'scale inside out' model. The proposed method exploits the usage of two components such as verification module and elastic load balancer in order to detect and mitigate the DDoS attack commendably.This enhanced method optimally reduces the resource utilization factor through elastic load balancing by analyzing the incoming traffic data and server condition. The main idea of scaling down the resources and services is to sacrifice the victim's resources during an attack period in order to mitigate the effect and recover from the attack. The performance of the proposed model is evaluated using various metrics to analyze the attack detection time, attack reporting time, and attack mitigation time. The results show that the proposed method works well to mitigate the DDoS attack by consuming the minimum resources thereby decreasing the service time of the user

Keywords: Distributed Denial of Service, Scale Down/Up, Cloud Attacks, Elastic Load Balancing, Resource Utilization.

1. Introduction

As cloud computing provides various serviceswith remarkable benefits such as virtualization, high scalability, on-demand service, pay as per use and so on, many business organizations, enterprises, and individuals are migrating applications towards cloud computing environment (Zhang et al., 2010). This makes the cloud an attractive place for assaulter to attempt their attacks. However, as the scale of cloud computing is large, the possibilities for distributed denial of service (DDoS) attacks are higher than traditional network attacks (Chahal et al., 2019;Mahjabin et al., 2017). The strike force and range are much larger in a cloud environment. Due to the higher possibility of DDoS attacks in the cloud and their intensity, the result of the attack detection algorithm gets highly influenced. Thus, dealing with more intense attacks as well as handling large traffic in the cloud environment is more difficult and sometimes the algorithm may not provide suitable results during DDoS attacks.

Denial of Service

Denial of Service (Zhen et al., 2009; Mell et al., 2000) attack is a distributed, cooperative, large-scale DoS attack, in which most of the attackers use compromised computers on the Internet as "zombies" and initiate intensive "denial of service" requests to a specific target to achieve the purpose of exhausting its network resources and system resources, so, that it cannot provide services to frequently requested users. Hackers often use "zombie masters" botnets (ie Botnet), launching large-scale DDoS flood attacks in which Web servers and DNS servers are the most common targets.

Usually, DDoS attacks in the cloud environment are successfully performed by controlling multiple distributed servers and PCs through a joint attack platform that intends to send incomplete requests to one or more targets making a large amount of malicious traffic that consumes huge network bandwidth or system resources. As a result, the normal requests sent by legitimate users will get rejected (Vishwakarma and Jain, 2020). In traditional networks, protecting a sufficient number of machines in a room from infections and minimizing the attack intensity is still complicated, but in the cloud environment providing on-demand services, the attack can be successfully made by creating a powerful botnetwork, so that the intensity of DDoS attack in the cloud environment is more on the increase in the network (Alani, 2016).

For DDoS attacks with larger traffic against the cloud environment, responding quickly to detect them is very difficult for traditional detection algorithms. So, calculating resource utilization is the key notion to detect such higher traffic flooding DDoS attacks on the cloud environment. The idea is that the detection algorithm can

quickly detect attack packets and minimizes excessive consumption of system resources by attack groups. High resource conflict or resource contention can be identified by the "Resource Utilization Factor" for all incoming requests. High resource conflict occurs between the attacker and the legitimate user which leads to delayed service (Iyengar et al., 2014). The resources are easily available during the attack downtime which in turn helps in attack mitigation through attack absorption.

Load Balancer

Typically, a load balancer located between the client and the server; handles incoming network and application traffic and distributes traffic through several backend servers using different algorithms. By balancing application requests across multiple servers, a load balancer reduces individual server load. It prevents each application server from becoming a single point of failure thereby enhancing the overall availability and responsiveness of the application. Elastic load balancing is an automatic distributor of incoming application traffic across multiple virtual machines. It enables the service provider to achieve a greater level of fault tolerance by seamlessly providing the required amount of load balancing capacity to distribute application traffic. When elastic load balancing detects error-prone instances, it automatically reroutes traffic to error-free instances until the error-prone instances have been restored. Customers can enable elastic load balancing within a single zone of many multiple availability zones for more consistent application performance.

More devices are involved in initiating the DDoS in the cloud, however, it varies from simple DoS attacks to frequent and large DDoS attacks in which there occurs high resource conflict that results in delay in processing the legitimate user request. In situations of extreme resource conflict, achieving attack absorption and further mitigation are highly challenging. In previous works dynamic auto-scaling and "Scale inside out"(Somani et al., 2017)were performed to achieve attack absorption and mitigation by maintaining the resource utilization factor minimum. For each incoming request,the resource utilization factor varies, and the scale inside out approach maintains the resource utilization factor minimum thereby resource contention can be kept in control. Dynamic auto resource scaling is much costlier and cannot be adopted in real-time by the service provider with a low budget (Chieu et al., 2009). Thus in this work, the scale inside out mechanism is enhanced with elastic load balancing which minimizes and manages high resource contention by expanding the resource utilization factor which is also able to mitigate the attack towards the victim source and avoids subsequent service delay for the legitimate user. The model fixes a threshold for resource utilization factor and balances the incoming load by introducing elastic load balancing with scale down strategy.

Elastic load balancer (ELB) applies vertical scaling on resources allocated for the services to mitigate the attacks after detecting the presence of attacks.

The further sections of the paper are structured as follows. Section 2 describes the previous research carried out in preventing, detecting, and mitigating DDoS attacks. Section 3 describes briefly about an existing scale inside out model for mitigating DDoS attacks which is a base for the proposed work. Section 4 describes the proposed dynamic resource scaling with an elastic load balancing model in which the model components and its working procedure along with the algorithm pseudocode are also presented in subsection 4.1 and 4.2. Section 5 discusses the performance evaluation of the proposed method and comparison with the existing model followed by section 6 that presents the conclusion of the proposed work.

2. Literature Survey

The research on DDoS attack detection and prevention for cloud computing includes a solution called Cloud Traceback (CTB) to identify the source of the HTTP and XML denial of service attack (Chonka et al., 2011). The authors also introduced a cloud protector that uses backpropagation to detect and handle the attack traffic. Yu et al. (2014) presented a dynamic resource allocation strategy for DDoS data centres attacks in the cloud. The authors utilized idlecloud resources and copied enough intrusion prevention servers for achievingspeedinfiltering DoS attack flow.

Girma et al., (2015)analyzed the currentDDoS detection technology with different parameters along with the advantages and disadvantages of each model. The authors also proposed a hybrid statistical model that effectively mitigates DDoS attacks. Osanaiyeet al. (2015) detected DDoS attacks by analyzing the characteristics of TCP/IP packet headers that contain the source of the data packet. Liu et al., (2016) proposed a method that takes the frequency domain characteristics from the autocorrelation sequence of the network flow as a clustering feature and uses the BIRTH algorithm to find the abnormal flow of traffic.

Kim et al., (2006) proposed the PacketScore scheme which utilizes the Bayesian formula to calculate the score of data packets. If the computed score is lower than the fixed threshold, then the packet is identified as an attacker's data packet. However, as the threshold is analyzed and fixed that does not consider the intensity, it is not suitable for handling large traffic in a cloud environment DDoS attacks. With these ideas as a base, several scholars proposed a series of detection and defence methods of DDoS attacks in the cloud environment. Dou et al.,

(2013) proposed filtering based on the confidence (CBF, Confidence-Based Filtering) method that groups them according to data for determining the legality of the group. Shamsolmoali et al (2014) presented a model using datamining and neural network techniques to detect the DDoS attack. This model helps specifically to detect the TCP attacks. KDD Cup dataset was used and detection accuracy was evaluated.The main advantage is that it requires less storage while making the detection faster

Sahi et al. (2017) suggested a defence model that detects flooding DDoS attacks by dividing data into groups and establishing a blacklist to store the source IP of attack packets. Jeyanthi et al. (2013) proposed a deception detection algorithm for the detection of a large-traffic DDoS attack launched on a server in a cloud environment. Navaz et al (2013) combine entropy-based systems with anomaly detection systems to provide multi-level detection methods to detect hidden small traffic DDoS attacks. Though the method provides decent results for the DDoS attacks with large traffic in the cloud environment attack, the time it takes to complete the process is very slow. Although the researchers have proposed some detection algorithms for flooding DDoS attacks and arbitrarily DDoS attacks, the presents result with the lower speed in the cloud environment and the methods rarely consider the actual system that is subjected to different attacks. So, the research pertaining to the security of the cloudenvironment must detect both traditional flooding DDoS attacks and low the rate-based DDoS attack in which providing cloud users with a secured network environment is of greater significance.Some of the work usesthe characteristics of SDN in detecting DDoS attacks in cloud environments.

Wang et al (2015) proposed a DDoS attack mitigation architecture that detects the attack and provides a fast attack reaction. The author also suggested that SDN network technology helps in defending against DDoS attacks. A framework was suggested to detect and mitigate the DDoS attack effect attacks (Saravanan et al., 2019). It makes use of three screening tests to prevent the server from attacks and uses various constraints to detect the attacks. It uses two queues to mitigate the attacks. A general defence strategy that is not specific to any attacks was suggested for mitigating the attack using pushback and resource regulation. An algorithm that is based on the aggregate-based congestion control that can be applied to routers that prevent from bandwidth congestion attacks and resource consumption attacks was proposed (Wang, 2008). Similarly, a model that prevents and detects the flooding DDoS attacks using simple distance based measures applied on TTL values specified in the IP packets was suggested (Chapade et al., 2013). Kalliola et al. (2015) presented an architecture in the cloud that combines the normal traffic, external blacklist, and flexible capacity calls that realizes automated defence against flooding DDoS attacks, but it is difficult to detect a low-rate DDoS attack with small traffic.

From the analysis made from the literature, few of the methods are not suitable for handling large traffic in a cloud environment. Some other methods find difficult to handle low-rate DDoS attack with small traffic. Though few methods produce better result for both low-rate and high-rate traffic, the time taken by themto complete the entre process is very low. Thus the proposed method exploits the usage of two components such as verification module and elastic load balancer in order to detect and mitigate the DDoS attack commendably.

2.1 Scale Inside Out

The utilization of the resources depends on the time taken to complete the request. Scaling the capacity of VM is the essential step to estimate the total number of requests processed in a particular time. Scale inside out approach not only scales the capacity, it also performs internal application scaling to reduce the resource utilization factor.This idea was proposed by Somani et al., (2017). As an initial step, the problem of attack absorption and delay is first addressed by reducing the resource utilization factor. To process the client request, several steps are performed such as a request for connection, successful connection establishment, receiving the request, analyzing the type of request, processing the request, sending a response to the requested client, and so on. For processing the client request, various resources are also utilized like processor, memory etc in which all the steps in processing request impact on this utilization factor.The above-mentioned steps can be implemented at service providers or client-server architectures, however, in the case of DDoS attacks, numerous requests are launched towards a victim server leading to a heavy resource contention which affects the mitigation service and also adds up to the resource utilization which results in a delay while processing request to the legitimate user.

In this situation scaling of the resource is very costly and may not provide a better solution for resource contention. The aim of scale inside out is reducing the request processing by reducing the utilization factor R_{Factor} to increase the number of requests $N_{max}$and capacity of the victim server $VM_{Capacity.}$

$$\text{Resource utilization factor } R_{Factor} = VM_{Capacity} / N_{max} \quad (1)$$

This is achieved by skipping the service processing stepsfurther giving way to resource mitigation by blocking and dropping attack connections. Figure 1 depicts the phases in DDoS attack mitigation.

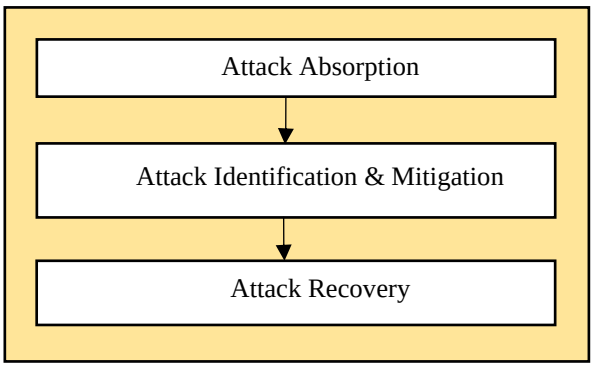


Figure 1. Phases in DDoS attack Mitigation

As the first phase, it identifies the attacker request targeting the victim server, next phase is the attack identification and mitigation where the attacker requests are identified and further connection establishment is stopped or only initial request is processed since the attacker intention is to deny the service given to the client, the third phase is recovering back to normal service processing as during attack the QoS is unchecked as priority is to serve the legitimate.In the scale-inside-out mechanism, applications are scaled internally to keep the resource utilization factor minimum thus solving delay in attack absorption, post-attack mitigation.

3. Proposed Dynamic Scaling With Elastic Load Balancing

The overall design of the proposed dynamic scaling with an elastic load balancing model is shown in Figure 2. The proposed idea has been inspired by the existing Anti-DDOS Framework (Saravanan & Sathya Bama, 2020) and Scale Inside-out, a DDoS mitigation framework (Somaniand Conti, 2017).

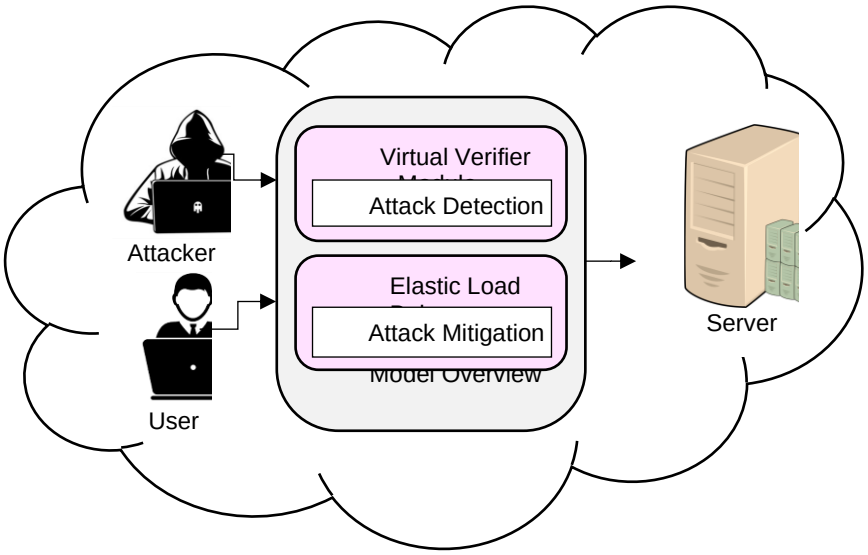


Figure 2. Overall Design of the Proposed ELB Model

The model has two main components namely verifier module (VM) (Sqalli et al., 2011) and elastic load balancer (ELB) to detect and to mitigate the attack by scaling down the resources as well as some services available to the server. The detailed framework of the proposed dynamic scaling with the ELB model is depicted in Figure 3.

In the proposed model, each request sent by the user or attacker is initially verified by the verifier module(VM) which protects the server as a shield from the requests intended for attacks. The verifier moduleapplies various verification analyses on the incoming requests such as incoming and outgoing packet statistics, analysis on protocols, number of connections, and number of packets with SYN on. Based on the analysis made on the characteristics of the requests, the DDoS attack will get detected. The result of the request verification phase is then forwarded to the elastic load balancer.

Elastic load balancer (ELB) applies vertical scaling on resources allocated for the services to mitigate the attacks after detecting the presence of attacks. Usually, for handling any request, most essential resources such as processor P, memory M, disk D, and Network Throughput (T) are sufficiently allocated for servicing their users on virtual machines. The number of connections to be established at a particular point in time varies based on the

allocated resources. The idea is to minimize the resources (scaling down) allocated to their users on detecting the attacks in order to mitigate the effects. Also, the requests that are initiated but not serviced completely due to more response time are also scaled down by withdrawing the allocated resources. Once the attacks are mitigated and recovered, then the resources are allocated normally (scaling up). Thus, ELB has the ability to scale down or scale up the minimum resource utilization factor. In general, elastic load balancing is an idea in which it can adapt to various fluctuations that occurred in the patterns of the network traffic promptly. Furthermore, auto scaling properties on the resources can also be implemented that guarantees adequate server capacity for varying levels of application load without demanding manual scaling of resources. The prominent two components such as verifier moduleand elastic load balancer are explained with their algorithm pseudocode in below sub sections.

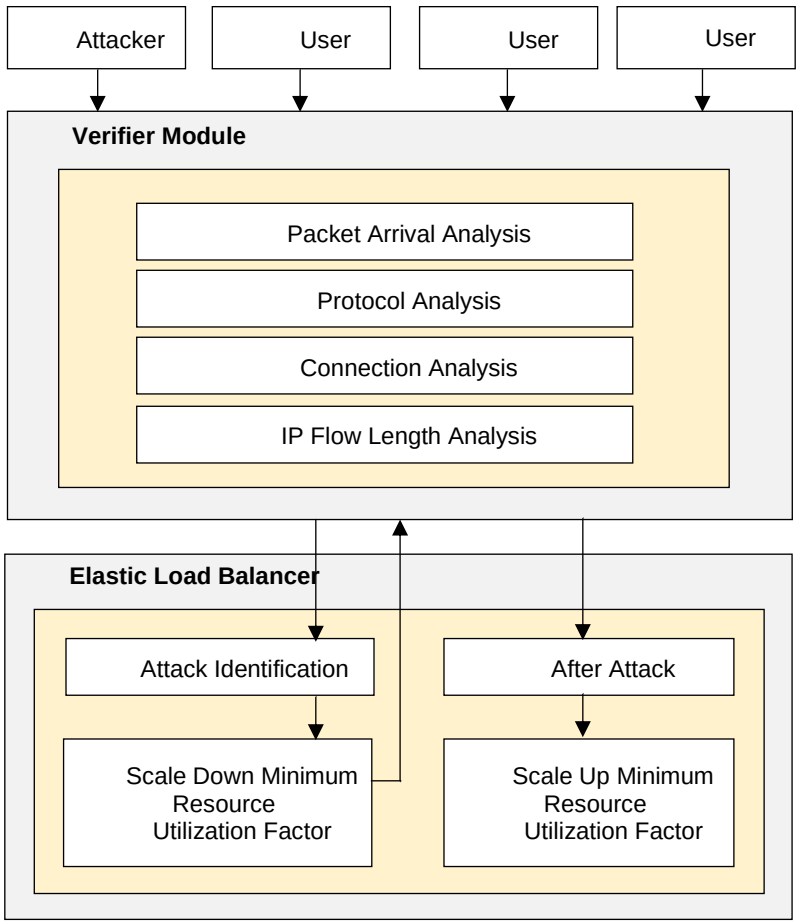


Figure 3. Framework of the Proposed Dynamic Resource Scaling with ELB Model

3.1 Attack Detection with Verifier Module

Verifier Module(VM) verifies the incoming requests on various factors. Several factors are available in the literature for analyzing the incoming packets (Suresh and Anitha, 2011). In order to attempt the DDoS attacks on any node, the attacker mainly utilizes flooding attacks in which the requests will follow some pattern concerning the packets or protocols. On analyzing these patterns, the attack can be detected clearly.

3.1.1 Analysis of PacketArrival

Generally, the packets will have several details in their header in order to identify the source and destination. The analysis made on the packet header ensures the detection of DDoS attacks. Examining the source address of the packets is the critical part in which the arrival of several request packets from the same source continuously indicates the possibility of DDoS attacks. Thus, if the number of requests from the same source sSR_{normal}in a particular timestamp exceeds the maximum number of requests from the same source sSR_{limit}, then there is a possibility of DDoS attack in which case the other analysis can be carried out. Similarly, the number of new requests from different sources is also another parameter to be considered in detecting attacks. Accordingly, if the number of new requests from different sourcesdSR_{normal} in a particular timestamp exceeds the maximum number of new requests from different sourcesdSR_{limit}, then there is a possibility of DDoS attack in which case the other analysis can be carried out.

On the other hand, the ratio of incoming requests to the outgoing requests is another significant parameter to be considered in detecting DDoS attacks. As the attacks are initiated by flooding the request to make the node to hang up completely on service overload, the number of incoming requests and the outgoing services are always analyzed (Saravanan & Sathya Bama). The proportion of the number of incoming requests and the number of outgoing services at a period of timestamp t is given in Eq. (2)

$$PR_{io} = \frac{n(Incoming\ IP\ Packets)_t}{n(Outgoing\ IP\ Packets)_t} \quad (2)$$

Logically, the proportion will always remain approximate constant if there is a normal service. However, if there is a sudden increase in the proportion value, then it highly indicates the event of an attack. Even during the high workload, the ratio of incoming and outgoing packet can show the attack presence. Logically, the incoming and outgoing packets will be balanced in the network during the normal traffic. Thus, if the ratio is less than are equal to 1, then it is a normal traffic and if the ratio is greater than 1, then the network traffic can be identified as attacks (Devi et al., 2019).

3.1.2 Analysis of Protocol

Similar to ratio analysis on incoming and outgoing packets, ratio analysis on various protocols such as TCP (T), UDP (U), and ICMP (I) packets are also analyzed. The denial of service can be initiated by sending various requests with various protocols due to which there will be a sudden increase in the proportion. Thus, the ratio of incoming packets with the three protocols indicates the attacks and it is computed as given in Eq. (3). For simplicity, the protocol proportions are specified in vector representation.

$$\langle PP_T, PP_U, PP_I \rangle = \langle \frac{\sum P_T\ Packets}{\sum IP\ Packets}, \frac{\sum P_U\ Packets}{\sum IP\ Packets}, \frac{\sum P_I\ Packets}{\sum IP\ Packets} \rangle \quad (3)$$

Apart from using the protocol proportions, the entropy value is also used to detect the arrival of attack packets. Generally, the entropy value of protocols remains non zero constant with the approximate value 0.43 (Xu et al., 2007) since the proportions are constant. On the other hand, during the DDoS attack, due to flooding of packets, the entropy value may attain 0. Thus, the entropy of protocols plays a significant role in detecting the attack. The formula to compute the entropy value of the protocols TCP, UDP, and ICMP is given in Eq. (4).

$$E_p = -PP_T log_2 PP_T - PP_U log_2 PP_U - PP_I log_2 PP_I \quad (4)$$

While discussing the protocols, flooding attacks initiated with the protocol is a major concern. Generally, the attackers use the SYN flag in the TCP packet for making the DDoS attack successful. Mostly, the attacker sends the packets continuously with the SYN flag on. If the count of the packet with the SYN flag onrepresented as P_{SYN} is greater than the C_{SYN} which is a limit set as a threshold, then there is a possibility of an attack. This can be extended to other flooding attacks with ACK flag value too.

3.1.3Analysis of Connection

Each virtual server machines are allocated with the various essential resources such as processor P, memory M, disk D, and network throughput (T). Let the resources allocated for the VM is represented as a vectoras in [Somani, & Conti,2017].

$$Res = \langle P, M, D, T \rangle \quad (5)$$

As the VMs have an ability to process the requests in parallel and the total resources are partitioned and are allocated to the simultaneous requests. However, the number of requests (n) to be processed parallel by the victim service depends on the demand of resources for each request. Similarly, the processing of the request may also require one or more resources. Thus the resource utilization factor can be defined for n requests as

$$Res_n = \langle P_n, M_n, D_n, T_n \rangle \quad (6)$$

With this information, the maximum number of requests that can be processed by the victim service can be computed as

$$NR_{max} = \frac{Res}{Res_n} \quad (7)$$

Here the number of requests depends on the number of connections established. Thus the maximum number of connections NC_{max} that can be established is the maximum number of requests NC_{max}that can be processed simultaneously. Thus

$$NC_{max} \leq NR_{max} \quad (8)$$

Thus, if the condition is reversed, that is, if the number of connections becomes greater than the maximum number of requests NC_{max}, it indicates that the attack has been initiated in which case it can be reported to the ELB or else further analysis can be carried out.

3.1.4 Analysis of IP Flow Length

Similar to protocol analysis, average length of IP flow is also another significant criterion to be taken for analysis to identify the attack packets. More generally, IP flow is considered as the number of packets belonging to the same criteria. The criteria include source IP address, destination IP address, source port, destination port and protocol used. These criteria are used to recognize the packet flow distinctively. Firmly, packets having same source and destination addresses, same source and destination address but with different protocols such as TCP, UDP and ICMP arriving sequentially belongs to the same packet flow. Thus, the length of the IP flow is the number of packets belonging to the same IP flow and it is calculated for the specific time interval. The average length of the IP flow $LIPF_{avg}$for a particular time period t is computed as in Eq. (9).

$$LIPF_{avg\_t} = \frac{\sum IP\_Packets}{\sum IP\_Flow} \tag{9}$$

Normally, the value for the average length of the IP flow lies between 5 to 10. On the other hand, the value falls close to 1 indicates the attack packets.

Apart from this, the entropy value of average length of the IP flow is also used to detect the attack packets. Generally, the entropy value of protocols lies between 2 and 4 approximately during normal packet flow but lies between 8 to 10 approximately during attack flow (Xu et al., 2007). The formula to compute the entropy value of average length of the IP flow for various protocols TCP (t), UDP (u), and ICMP(i) is given in Eq. (10).

$$E_{Ip\_flow} = -p_T log_2 p_T - p_U log_2 p_U - p_I log_2 p_I \tag{10}$$

Here *p*indicates proportion of the length of IP flow for a period of time t to the number of IP packets for each protocol.

The algorithm to detect the attack by the verifier module is given in Figure 4.

Algorithm: Pseudocode for Verifier Module
Input: IP address of Packet P, Threshold values
sSR_{limit} - Maximum number of requests allowed from the same source
sSR_{normal} - Number of requests from the same source
dSR_{limit} - Maximum number of new request allowed from different sources
dSR_{normal} - Number of a new request from different sources
NR_{max} - Maximum number of requests that can be processed by the victim
C_{SYN} - Maximum number of packets allowed with SYN flag set
NC_{max} - Number of connections
pac_{in} and pac_{out} – Number of incoming and outgoing packets
Output: Detection of attack
*Algorithm **VVM_procedure()***
Begin
//Packet Analysis
Compute sSR_{normal}, $dSR_{normal,}$
Compute proportion of incoming and outgoing requests $PR_{io} = pac_{in} / pac_{out}$
IfsSR_{normal}<sSR_{limit}&&dSR_{normal}<dSR_{limit} && PR_{io}≤ 1
    Move to Protocol Analysis Block
Else
    //Attack detected and forward to Elastic Load Balancer
    Call ELB_procedure()
// Protocol Analysis
Compute protocol proportions PP_T, PP_U, PP_I as given in Eq. (3)
Compute Entropy of protocols E_p as in Eq. (4)
If deviation is low in PP_T, PP_U, PP_I&& E_p≠ 0 && P_{SYN} <C_{SYN}
    // Connection Analysis
    If NC_{max} < NR_{max}
     Move to IP Flow Analysis Block
Else
    //Attack detected and forward to Elastic Load Balancer
    Call ELB_procedure()
// IP Flow Analysis
Compute $LIPF_{avg}$ as given in Eq. (9)
Compute Entropy of length of IP flow E_{lp_flow} as in Eq. (10)
If 5<$LIPF_{avg}$ <10 && 2<E_{lp_flow}<4
    //Accept the packets and forward to Elastic Load Balancer
    Call ELB_procedure()
Else
    //Attack detected and forward to Elastic Load Balancer
    Call ELB_procedure()
End Function

Figure. 4. Algorithm for Detect the Attack

3.2 Attack Mitigation with Elastic Load Balancer

In the previous phase, the verifier module verifies the requests and finds whether the attack has been initiated. The result of this analysis report is then forwarded to the elastic load balancer. Initially, the resources are allocated favourably to the services. If the analysis identifies the occurrence of a DDoS attack, then the elastic load balancer scales down the resource utilization factor immediately thereby reducing the resources allocated to the services with the aim of mitigating the attack. Here the minimization of resource utilization factor Res_n to Res_{attack}which indicates the reduction in resources allocated to the service during the attack mitigation period. In the proposed work, Res_{attack} is assigned with a threshold value, however, it can be allocated with the dynamic values based on the intensity of an attack. Here, instead of releasing the resources completely, it can be released one by one after mitigation of attack, that is, the stage that is suspended can be resumed back one by one instead of releasing them

completely. Minimizing the resource utilization factor during the attack and bouncing back to normal at the end of the attack with the help of elastic load balancing results in an increase in capacity of the virtual machine. Scaling down the resource utilization factor is made by skipping the phases that include the utilization of resources by the services such as processor usage, memory usage, disk usage, and throughput usage along with the request processing phase and response preparation phase [Somani and Conti, 2017].

Additionally, only the initial requests at the time of high traffic are serviced and the further requests are retransmitted. This implies that only the index page is shown to the users and further requests are held in a waiting state or retransmitted. Attackers who are launching a huge number of requests are not waiting for the service reply and so retransmitting such requests or just processing the initial requests may reduce the resource utilization factor value and increase the capacity of VMthus paving way for attack mitigation.

The main stages of providing service at the normal time are given in Figure 5.

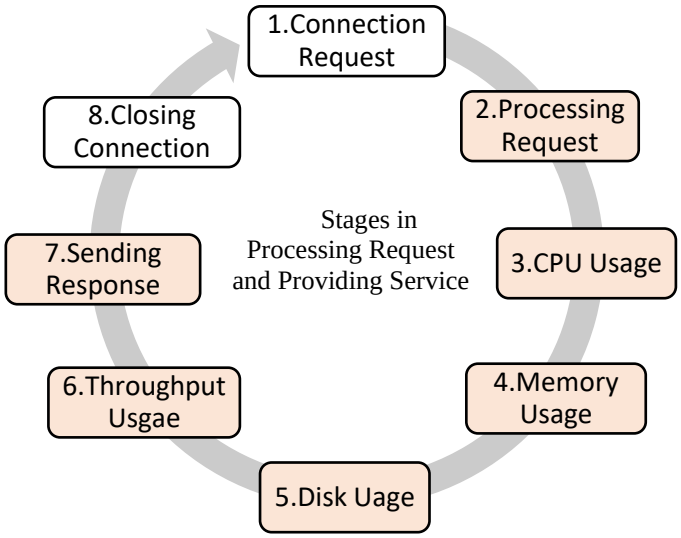


Figure. 5. Scaling Down the Resource during Attack

However, after identification of attack existence, few of the stages such as processing request, using resources such as CPU, memory, disk, and throughput, sending response are skipped in order to mitigate the attack. The stages presented in color in Figure 5 represents the skipped stages during attack existence. After attack recovery, the resources are scaled up which means that the service is allocated with normal resources and thus the service processes the request, uses the resources, and finally sends the response to the client.

The elastic load balancing using scale up/down method saves the victim server by adjusting the resource utilization factor Res_n minimum during the attack and bringing it back to normal during attack downtime thus it will not allow the targeted system to shut down or close completely and helps in retrieving the attacked data as the successful step of mitigation. The algorithm to mitigate the attack by scaling down the resources and services by elastic load balancer is given in Figure 6.

Algorithm: Pseudocode for Elastic Load Balancer
Input: Allocation of resources
Res_n – Resource utilization factor for the victim
Res_{attack} – Scaling down Resource utilization factor for the victim during the attack
Output: Attack Mitigation
*Algorithm **ELB_procedure()***
Begin
Allocate the resources to the web services
resource utilization factor = Res_n
Apply the resource utilization factor to the web service
Do
Call VNM_Procedure() periodically analyse the incoming packets
If attack detection is True
//Scale down the resource utilization factor by presenting the index page
resource utilization factor = Res_{attack}
Apply the new resource utilization factor to the web service
Else //attack detection is False
//Scale up the resource utilization factor as normal
resource utilization factor = Res_n
Apply the original resource utilization factor to the web service
End While
End

Figure. 6 Algorithm to Mitigate the Attack

Several threshold values have been used in order to identify the attacks. Some threshold values such as the number of requests allowed from the same source (sSR_{limit}) and the number of new requests arrived from different sources (dSR_{limit}) can be set by analyzing and learning the history of the user experience in the system. On the other hand, other threshold values such as number of connections (NC_{max}), number of requests allowed (NR_{max}), allocating the resources to the service (Res_n), and scaling down the resources after attack detection (Res_{attack}) can be computed based on the system capacity. The requests that are in the processing stage but that take more processing time after the attack detection are also stopped and the allocated resources are withdrawn with the aim of scaling down the services during the attack period. Thus the overall idea is to sacrifice the resources of the victim during an attack in order to mitigate the effect and to recover from the attack.

4. Experimental Analysis

To demonstrate the availability of the victim server to process the request even during the attack, an experiment is performed by sending 50 legitimate user requests, attacker traffic requests, SSH requests. These are launched towards the victim server at the same time. Cloudsim simulation tool is used to simulate the results. The victim server service is to convert the uploaded word document into pdf format. Intel Xeon processor is chosen as the victim server processor with Xen Hypervisor. Traffic rate is assumed between 100 to 1000 concurrent requests which will not exceed 1000. The configuration for set1 file conversion service is 8 vCPUs and 16GB, configuration for set2 file conversion service is 16 vCPUs and 32GB. Input files of various sizes are submitted to analyze the performance with varying file sizes.

The performance of the system is analyzed based on the attack detection made for the given inputs. It is analyzed with two metrics such as attack detection time and attack reporting time. The obtained value is examined by varying the file sizes and the outcomes are compared with the existing scale inside-out model [Somani et al., 2017]. The obtained results are presented in Table 1.

Table 1. Performance Comparison of DDoS Attack Detection

Resource Set	Resource Type	Attack Detection Time in seconds			Attack Reporting Time in seconds		
		Without Existing SIO	With Existing SIO	With Proposed Model	Without Proposed Model	With Existing SIO	With Proposed Model
Set 1	50 kB	39.52	37.12	37.12	38.16	37.1	36.14
	100 kB	40.87	38.14	37.14	41.14	38.11	40.44
	1 MB	42.3	40.77	39.45	42.4	40.22	41.16
	2 MB	45.23	43.45	42.66	44.76	41.01	37.66
Set 2	50 kB	42.38	40.03	39.23	42.03	40.03	39.38
	100 kB	42.33	40.03	39.26	41.67	40.03	39.33
	1 MB	42.56	41.03	40.55	41.27	41.03	38.34
	2 MB	43.61	42.02	41.24	43.44	42.07	37.03

The performance of the system is also analyzed by comparing the performance of the service provided by the server. The experiment is examined with two metrics such as service time and service downtime. The obtained values are analyzed with varying file sizes and the results are compared with the existing scale inside-out model (Somani et al., 2017). The obtained results are presented in Table 2.

Table 2. Performance Comparison of Victim Service

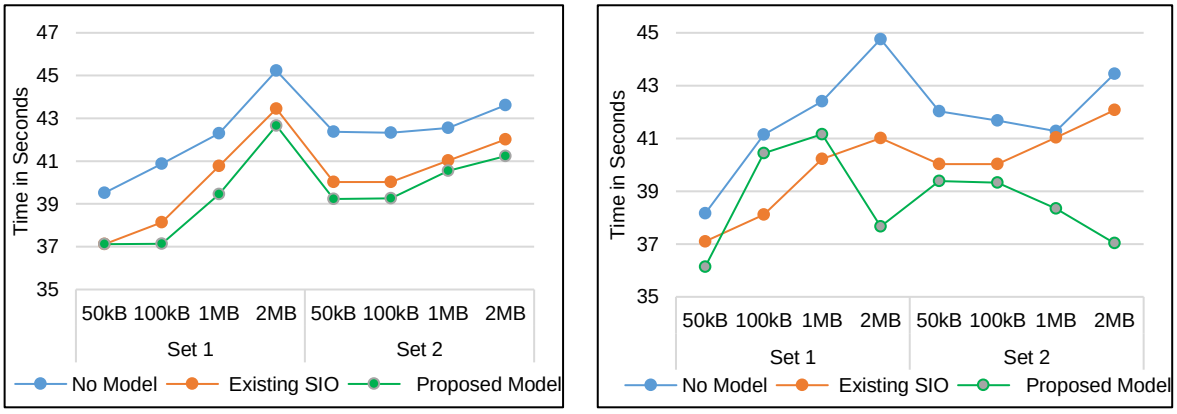
Resource Set	Resource Type	Victim Service Downtime in seconds			Time to Service Request in seconds		
		Without Existing SIO	With Existing SIO	With Proposed Model	Without Any Model	With Existing SIO	With Proposed Model
Set 1	50 kB	42	77	74	43.65	14.4	14.1
	100 kB	235	78	70	125.5	14.2	14
	1 MB	445	75	68	256.7	15.7	14.8
	2 MB	675	88	81	358.6	17.7	16.8
Set 2	50 kB	46	36	25	48.4	13.84	12.73
	100 kB	233	33	26	187.2	13.72	12.61
	1 MB	530	15	9	289.5	13.85	12.74
	2 MB	663	28	20	397.2	14.35	13.21

From Table 1 and Table 2 it is clear that the proposed model reaches the attack downtime faster than earlier approaches, thus the victim server recovers faster from attack and processes the requests of the legitimate user better than other ways of mitigation. High contention of resources occurs as the victim server is under high traffic, the following metrics are found before applying the elastic load balancing to scaling down like detection time of the attack, reporting time of the attack, downtime of service, and service recovery time. After applying the elastic load balancing, and performing the reduction in Res_n through scaling down the resources, the above metrics are again calculated such as detection time of the attack, reporting time of the attack, downtime of service attack, service recovery time all show a drastic reduction.

From Table 1, it is observed that the attack detection time is known only after the impacts of attacks are over. Once the victim server retains the service available state, attack reporting time and detection time can be observed. The higher resource conflict or resource contention results in the unavailability of VM connection establishment by not allowing the VM interface. However, after applying the proposed scalingdown the resources, service response time is much reduced showing better mitigation performance. Attack downtime of recent DDoS attacks may take a minimum of days to even weeks leading to service unavailability thereby creating a huge loss. In the proposed model, the resource utilization factor keeps varying, and during downtime, resource availability will be minimum and thus the proposed scheme will ensure the availability of possible resources for the mitigation process and extracts the attack data time-to-time. The results shown in table 1 and table 2 is depicted as a graph in Figure 7 and Figure 8.

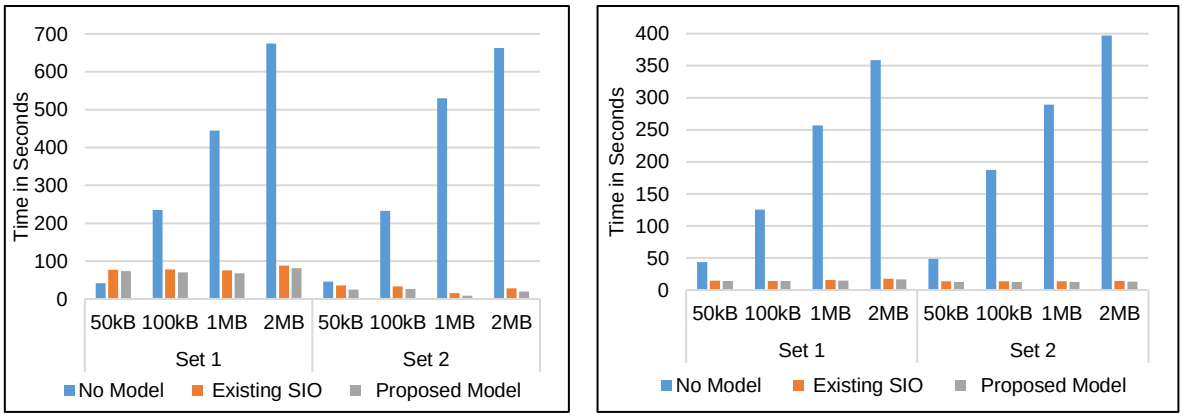
Performance evaluation has been analyzed by comparing the response time with a varying number of requests for the proposed model and the existing SIO model. The results are represented as a graph in Figure 9 in which Figure 9(a) represents the response time for 50kB data by varying the request from 0 to 400, (b) represents the

response time for 100kB data by varying the request from 0 to 400, (c) represents the response time for 1MB data by varying the request and (d) represents the response time for 2MB data by varying the request count.



a) Attack Detection Time Comparison b) Attack Reporting Time Comparison

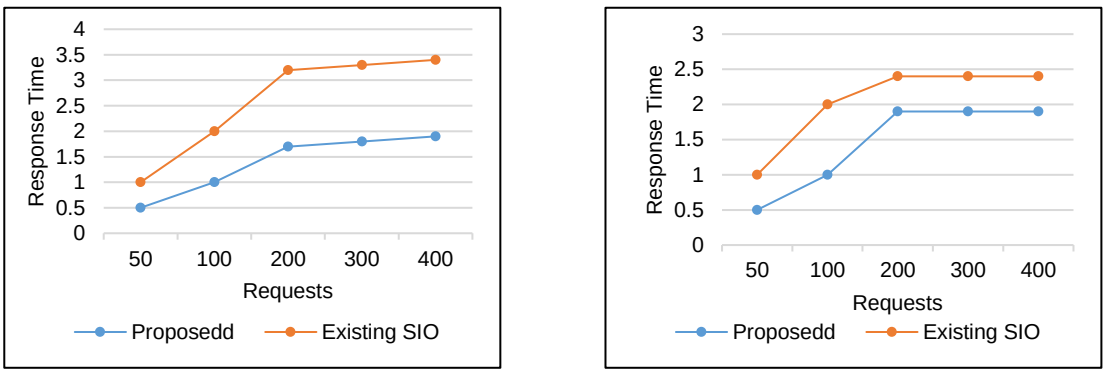
Figure 7. Performance Comparison of DDoS Attack Detection



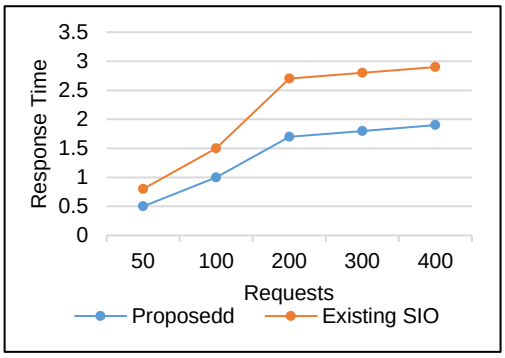
a) Victim Service Downtime b) Time to Service Request

Figure 8. Performance Comparison of Victim Service

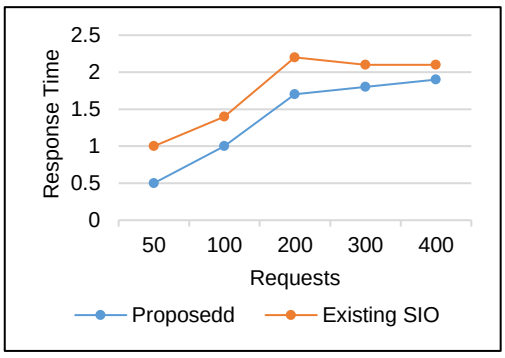
This research aims to speed up attack mitigation by reducing delay in attack identification. The attacker aims to create heavy traffic and to make the service unavailable and in turn, the attacker node will not wait for service response. Initially, during an attack, legitimate users may feel like the useless response is obtained from the server but after the mitigation process and attack downtime, the user will be properly serviced. Resource utilization factor will be reduced during the downtime. Hence to avoid loss and service unavailability, downtime is forced so that Res_n is reduced. Victim server at any cost will not shutdown, attack data are extracted and collected and after the mitigation and recovery server switches to service availability.



(a) 50kB service by varying request count (b) 100kB service by varying request count



(c)1MB service by varying request count



(d) 2MB service by varying request count

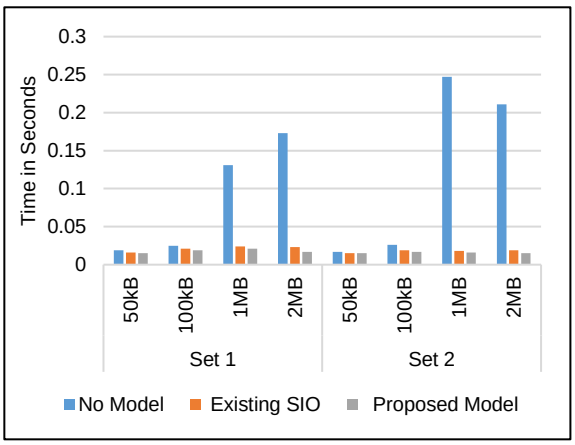
Figure 9. Performance Analysis on Response Time

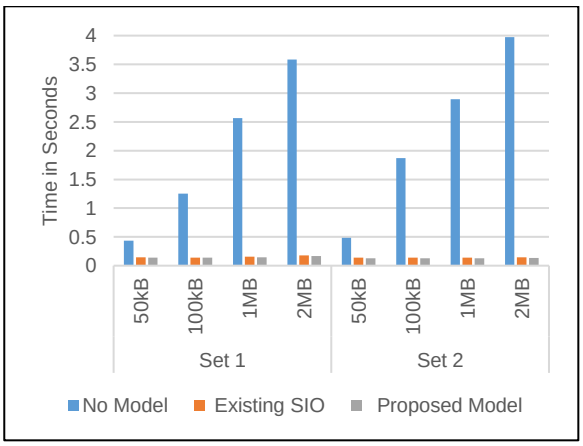
Thus, the performance of the system for mitigation service during attack time is compared with the existing SIO model by analyzing the maximum and minimum response time. The result obtained for the DDoS mitigation service is presented in Table 3.

Table 3. Performance Comparison of DDoS Mitigation Service

Resource Set	Resource Type	Maximum Response Time in Seconds			Minimum Response Timein Seconds			Average Response Timein Seconds		
		Without Any Model	With Existing SIO	With Proposed Model	Without Any Model	With Existing SIO	With Proposed Model	Without Any Model	With Existing SIO	With Proposed Model
Set 1	50 kB	6.145	2.098	2.043	0.019	0.016	0.015	0.437	0.144	0.141
	100 kB	112.47	2.014	1.945	0.025	0.021	0.019	1.255	0.142	0.140
	1 MB	231.21	2.141	2.051	0.131	0.024	0.021	2.567	0.157	0.148
	2 MB	379.23	2.273	1.456	0.173	0.023	0.017	3.586	0.177	0.168
Set 2	50 kB	6.236	2.114	2.027	0.017	0.015	0.015	0.484	0.138	0.127
	100 kB	107.86	2.044	1.746	0.026	0.019	0.017	1.872	0.137	0.126
	1 MB	212.57	2.579	2.457	0.247	0.018	0.016	2.895	0.139	0.127
	2 MB	334.56	2.145	1.311	0.211	0.019	0.015	3.972	0.144	0.132

Here the average maximum response time for the given data is 2.17 seconds for the existing method whereas it is 1.87 seconds for the proposed method. Similarly, the average minimum response time for the existing method is 0.02 which is approximately the same for the proposed model. This shows that attack mitigation is fast for the proposed model than the existing model. The values presented in table 3 for minimum and average response time is depicted as a graph in Figure 10.





(a) Minimum Response Time (b) Average Response Time

Figure 10. Performance Comparison of DDoS Mitigation Service

The number of attacks processed before detecting time is another significant metric to be used for evaluating the system. An increase in the number of attacks serviced unknowably will decrease the performance and speed of the system. Thus, the existing SIO model and proposed model has been compared for various input size and the results are shown in Table 4.

Table 4. Count of Attacks Processed before Detection

Input Set	**Input Size**			
	50kB	**100 kB**	**1MB**	**2MB**
Set 1 (No model)	4098	631	15	7
Set 1 (SIO)	175	41	12	9
Set 1 (Proposed Model)	131	27	11	10
Set 2 (No model)	6871	1571	37	18
Set 2 (SIO)	371	129	27	11
Set 2 (Proposed Model)	312	117	24	12

The total number of attacks serviced undetectably for the proposed model is 179 with set 1 and 465 for set 2, whereas the count is 237 for set 1 and 538 for set 2. This shows that the proposed method acts quickly in detecting the attack as soon as possible. Here the existing SIO method provides service for 4.98% of attack for set 1 and 6.33% of attack for set 2 before detecting them. However, the proposed model serves 3.76% of attacks for set 1 and 5.47% of attacks for set 2 before attack detection. This shows that the proposed method is 1.22% and 0.86% faster for set 1 and set2 in detecting attacks than the existing model. This result shows that the proposed model detects the attack faster than the existing method.

The major finding of the proposed work is that increasing the resource of the VM does not increase the performance of the VM. This can be clear from the results shown in Table 1, Table 2 and Table 3 in which the results for the set 1 and set 2 are more similar with minimum difference in detecting and reporting the attacks, attack downtime and overall service time of the requests. The performance of the system in exploiting resources without congestion depends on various factors such as application type, input type, input size and utilization factor.

The another main thing to be noted is that conflict of resources occurs when the attack is not detected on time. Thus, using multiple analysis in detecting attacks such as packet analysis, protocol analysis, connection analysis and IP flow length highly increases the detection rate. However, the analysis has to be made periodically for effective results. This clearly shows that increase in the frequency of detection analysis (VVM) highly increases the detection rate. However, increased detection rate decreases the resource conflicts there by increasing the system performance in detecting attacks. This method results in high attack detection rate and thus the number of attacks processed before detection is very low when compared with other methods.

$$n(exec(VVM\_procedure)) \propto attack\ detection\ rate$$

$$attack\ detection\ rate \propto \frac{1}{resource\ conflicts} \propto performance$$

On the other hand, scaling down the resources after attack detection is most significant step to mitigate the attacks which in turn increases the overall performance of the system in mitigating attacks.

$$exec(ELB\_procedure) \propto resource\ scaling$$

$$resource\ scaling \propto mitigation\ rate \propto performance$$

The resource utilization factor which is mentioned earlier is kept in the normal state before the attack as Res_n, in which the cloud server can process n requests approaching it. The resource utilization factor depends on the number of requests and the capacity of the processor directly. During the attack, the resource utilization factor is aimed to reach a minimum as Res_{attack}, so that the resources are more utilized for mitigation service thus maintaining the Res_{attack} aslow and forcing the attack downtime to make the server recover back to service from the contention.

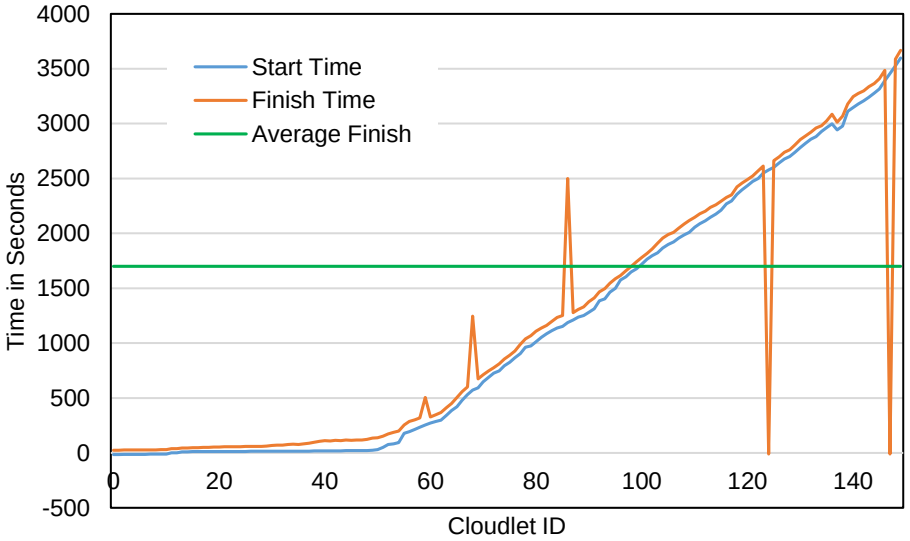


Figure11. Execution time in Seconds

Figure 11 illustrates the execution time of the request by the legitimate user in the proposed approach and the load is balanced by Sthe resource and scaling the service provided to the user having more response time. The peaks are due to attack mitigation where resources are utilized for mitigating and not for services. During an attack, the service may be slower but the server might not shut down leading to any loss after reaching the downtime of the attack everything gets back to normal and the user requests are processed immediately.

5. Conclusion

In client-server-based cloud architecture, the frequent repetitive and massive DDoS attacks result in high resource contention targeting a victim server which may result in collateral damages and heavy resource conflict resulting in service delay for a legitimate user. To address this problem previously autoscaling of the resources is dynamically performed during DDoS attacks which is a good approach but not cost-effective. Thus, to aid the low budget service provider to save from such attacks by mitigating DDoS attacks, a dynamic scale down mechanism is performed which provides a better result in mitigating DDoS attacks by reducing the "resource utilization factor" through which the capacity of the victim is increased. The model used a verifier moduleto detect the attack and on successful detection of an attack, scaling down the resources and services are specifically performed by elastic load balancer which implements the attack mitigation procedure by minimizing the resource utilization factor. However, after the successful mitigation of attacks, theoriginal resources are allocated to the victim service in order to adjust the load back. This model reduces the attack downtime, thus avoiding resource contention and giving way for mitigating DDoS attacks optimally and cost-effectively. The future work intends to extract the zombies automatically using machine learning capabilities by inheriting the network characteristics. Also, the spatiotemporal characteristics can be incorporated while designing the behavior defense mechanism

References

1. Zhang, Q., Cheng, L. and Boutaba, R., 2010. Cloud computing: state-of-the-art and research challenges. Journal of internet services and applications, 1(1), pp.7-18.
2. Chahal, K. J., Bhandari, A. and Behal, S., 2019. Distributed Denial of Service Attacks: A Threat or Challenge. New Review of Information Networking, 24(1), pp.31-103.
3. Mahjabin, T., Xiao, Y., Sun, G. and Jiang, W., 2017. A survey of distributed denial-of-service attack, prevention, and mitigation techniques. International Journal of Distributed Sensor Networks, 13(12), p.1550147717741463.
4. Vishwakarma, R. and Jain, A.K., 2020. A survey of DDoS attacking techniques and defence mechanisms in the IoT network. Telecommunication Systems, 73(1), pp.3-25.
5. Alani, M.M., 2016. Elements of cloud computing security: A survey of key practicalities. Springer International Publishing.
6. Iyengar, N.C.S.N., Ganapathy, G., Mogan Kumar, P.C. and Abraham, A., 2014. A multilevel thrust filtration defending mechanism against DDoS attacks in cloud computing environment. International Journal of Grid and Utility Computing, 5(4), pp.236-248.

7. Chieu, T.C., Mohindra, A., Karve, A.A. and Segal, A., 2009, October. Dynamic scaling of web applications in a virtualized cloud computing environment. In 2009 IEEE International Conference on e-Business Engineering (pp. 281-286). IEEE.
8. G. Somani, M. S. Gaur, D. Sanghi, M. Conti and M. Rajarajan, "Scale Inside-Out: Rapid Mitigation of Cloud DDoS Attacks," in IEEE Transactions on Dependable and Secure Computing, vol. 15, no. 6, pp. 959-973, 2017.
9. Chonka A, Xiang Y, Zhou W L. (2011) Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks. Journal of Network and Computer Applications, 34(4),1097-1107.
10. Yu S, Tian Y H, Guo S. (2014) Can we beat DDoS attacks in clouds?. IEEE Transactions on Parallel and Distributed Systems, 25(9):2245-2254.
11. Girma A, Garuba M, Li J. (2015) Analysis of DDoS attacks and an introduction of a hybrid statistical model to detect ddos attacks on cloud computing environment,12th International Conference on Information Technology-New Generations,212-217.
12. Osanaiyaeo A, Dlodlo M. (2015) TCP/IP header classification for detecting spoofed DDoS attack in Cloud environment, International Conference on Computer as a Tool. 1-6.
13. Liu Z G, Yin X C, Lee H J. (2016) A new network flow grouping method for preventing periodic shrew DDoS attacks in cloud computing,18th International Conference on Advanced Communication Technology (ICACT), 66-69.
14. Kim Y, Lau W C, Chuah M C. (2006) PacketScore: a statistics-based packet filtering scheme against distributed denial-of-service attacks,IEEE Transactions on Dependable & Secure Computing, 141-155.
15. Dou W, Chen Q, Chen J. (2013) A confidence-based filtering method for DDoS attack defense in cloud environment. Future Generation Computer Systems, 1838-1850.
16. Shamsolmoali P, Alam M A, Biswas R. (2014) high rate DDOS filtering method in cloud computing, International Journal of Computer Network & Information Security, 43-50.
17. Sahi A, Lai D, Li Y. (2017) An efficient DDoS TCP flood attack detection and prevention system in a cloud environment. IEEE Access,6036-6048
18. Jeyanthi N, Barde U, Sravani M. (2013) Detection of distributed denial of service attacks in cloud computing by identifying spoofed, International Journal of Communication Networks & Distributed Systems, 262-279.
19. Navaz A S S, Sangeetha V, Prabhadevi C. (2013) Entropy based anomaly detection system to prevent DDoS attacks in cloud, International Journal of Computer Applications, 42-47.
20. Wang B, Zheng Y, Lou W. (2015) DDoS attack protection in the era of cloud computing and software-defined networking. Computer Networks, 81(C): 308-319.
21. Saravanan, A., Bama, S.S., Kadry, S. and Ramasamy, L.K., 2019. A new framework to alleviate DDoS vulnerabilities in cloud computing. International Journal of Electrical & Computer Engineering (2088-8708), 9.
22. Shamsolmoali, Pourya, M. Afshar Alam, and Ranjit Biswas, "C2DF: High Rate DDOS filtering method in Cloud Computing.", International Journal of Computer Network and Information Security 6.9 pp: 43, 2014.
23. Wang, X., "Mitigation of DDoS Attacks through Pushback and Resource Regulation", In International Conference on MultiMedia and Information Technology, IEEE, pp. 225-228, 2008.
24. Chapade, S.S., Pandey, K.U. and Bhade, D.S., "Securing cloud servers against flooding based DDoS attacks". In International Conference on Communication Systems and Network Technologies (CSNT), IEEE, pp. 524-528, 2013.
25. Kalliola A, Lee K, Lee H. (2015) Flooding DDoS mitigation and traffic management with software defined networking, International Conference on Cloud Networking.248-254.
26. Zhen X U, JIN M A, SONG L U. (2009) Research on DOS Attack and Defense of SIP, Information Security & Communications Privacy, (1):37-42. 43
27. Mell P, Marks D, Mclarnon M. A. (2000) Denial-of-service Resistant Intrusion Detection Architecture, Computer Networks the International Journal of Computer & Telecommunications Networking, 34(4): 641-658.
28. A.Saravanan, S.Sathya Bama (2020) Multi-Model Anti-Ddos Framework For Detection And Mitigation Of High Rate Ddos Attacks In The Cloud Environment, International Journal Of Scientific & Technology Research, Volume 9, Issue 03, pp.4503-4511.
29. M.H. Sqalli, F. Al-Haidari, and K. Salah, K., "EDoS-shield-a two-steps mitigation technique against EDoS attacks in cloud computing," Proc. of IEEE fourth International Conference on Utility and Cloud Computing, pp. 49-56, 2011.
30. M. Suresh, and R. Anitha, "Evaluating machine learning algorithms for detecting DDoS attacks," International Conference on Network Security and Applications, Springer, Berlin, Heidelberg, pp. 441-452, 2011.

31. Devi, B.K., Dudeja, K., John, A.V., Marcin, K. and Subbulakshmi, T., 2019. Symmetric Measure of Network Traffic using Packet Ratio and Packet Symmetry. Procedia Computer Science, 165, pp.112-118.
32. T. Xu, D. He, and Y. Luo, "DDoS attack detection based on RLT features," in Proc. of IEEE International Conference on Computational Intelligence and Security, pp. 697-701, 2007