Computer Networks 187 (2021) 107825

Contents lists available at ScienceDirect

Computer Networks

journal homepage: www.elsevier.com/locate/comnet

The impact and mitigation of ICMP based economic denial of sustainability attack in cloud computing environment using software defined network

Sayed Qaiser Ali Shah [a], Farrukh Zeeshan Khan [a], Muneer Ahmad [b,*]

[a] Department of Computer Science, University of Engineering and Technology, Taxila, Pakistan
[b] Department of Information Systems, Faculty of Computer Sciences and Information Technology, University of Malaya, 50603 Kuala Lumpur, Malaysia

ARTICLE INFO

Keywords:
EDOS
ICMP detection and mitigation model (EDOS-IDM)
Cloud computing environment
Software defined network (SDN)
Auto-scaling
ICMP flooding attack
DoS/DDoS

ABSTRACT

High availability in network services is a crucial requirement for quality of experience. Denial of Service (DoS) and Distribute Denial of Service (DDoS) attacks are under contemplation by many researchers across the globe because these attacks directly target services availability. For this reason, cloud providers use the auto-scaling feature in Cloud Computing Environments (CCE), in which cloud resources scale dynamically on demand. DoS/DDoS attacks on CCE, using auto-scaling, do not deny services but cause high resource usage and substantial financial damages that become an Economic Denial of Sustainability (EDOS) attack. One of the DoS/DDoS attacks, resulting EDOS attack is the Internet Control Messaging Protocol (ICMP) flooding attack. In this paper, a novel technique, ICMP detection and mitigation model (EDOS-IDM) is proposed that can detect and mitigate Volumetric and Normal Behavioral ICMP traffic attacks. The results from the proposed technique are compared with the Normal Behavioral ICMP traffic attack because it causes least resource usage among all the mitigation techniques. According to our study, there is no such technique that can handle normal behavioral ICMP traffic attack. The technique is practically tested and evaluated on OpenStack production Cloud Environment test bed. According to the results, the technique is proved to save extra resource consumption and customer's bills in a cloud computing environment.

1. Introduction

Cloud computing (CC) is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. CC becomes the most powerful tool to deliver data over the internet. Using web-based tools, data can be accessed anywhere, anytime [1].

Security is one of the one of the major components of concern in the Cloud. According to Bruce Schneier, "the only secure computer is one that's turned off, locked in a safe, and buried 20 feet down in a secret location- and I'm not completely confident of that one either" [2]. The cloud environment is accessible online, therefore, it is prone to intrusion and unauthorised activity over a network. Detecting an intrusion depends on the defender's knowledge about attacks. According to a survey [2] extortion, identified revenge, political issues, competition between cloud providers, and proficiency testing by cybercriminals are common causes of intrusion resulting in DoS and DDoS attacks. These types of attacks are significant threats to the cloud computing environment (CCE). Some of the most complex DDoS attacks targeting CCE are HTTP Flood, Ping ICMP Flooding Attack, Network Time Protocol (NTP) Amplification attack, TCP SYN Flooding Attack, UDP Flooding Attack, Domain Name System (DNS) Amplification Attack, DNS Flooding Attack, Simple Service Discovery Protocol (SSDP) Amplification Attack, TCP Fragmentation, TCP Connection Flood, TCP RST Attack and TCP SYN-ACK Attack [3,4].

ICMP flooding attack is an easy to launch attack. In this type of DoS/DDoS attack, the attacker sends ICMP echo-request packets to overwhelm a targeted device and causing the target inaccessible to legitimate traffic. Today's cloud computing (CC) service providers offer auto-scaling feature where resources dynamically scale according to the customer's requirements and clients are charged as pay-per-use against CPU, memory, storage and network bandwidth [5]. DoS/DDoS attacks in auto-scaling do not make the resources unavailable to clients but lead to an attack where the client's cloud resource consumption is excessively high and assaults significant financial damage to the clients. This type of attack was named in November 2008 by Cristofer Hoff as "Economic

* Corresponding author.
E-mail addresses: qaiser.ali@uettaxila.edu.pk (S.Q. Ali Shah), farrukh.zeeshan@uettaxila.edu.pk (F. Zeeshan Khan), mmalik@um.edu.my (M. Ahmad).

https://doi.org/10.1016/j.comnet.2021.107825
Received 18 September 2020; Received in revised form 5 December 2020; Accepted 4 January 2021
Available online 7 January 2021
1389-1286/© 2021 Elsevier B.V. All rights reserved.

Table 1
Comparative summary of DDoS Mitigation Techniques.

Approach	Efficiency	Adaptive	Overhead	Scalability issues	Overfitting
Signature-based			✓	✓	
Artificial Intelligence	✓	✓		✓	✓
Machine Learning	✓	✓	✓		
Classifier	✓	✓	✓		
Data Mining	✓				
Statistical	✓				
Hybrid	✓	✓	✓		

Denial of Sustainability" (EDOS) [6]. The attacks on cloud computing environment are network-based attacks, therefore, there is an immense need of network-based protection techniques. Software-Defined Networks (SDN) is an alluring platform to mitigate network-based EDOS attacks [7]. SDN is a cost-efficient technology for both cloud service users and providers. SDN separates the control plane from the data forwarding plane [8]. For the control plane, a centralized controller can be used that takes decisions based on the pattern or parameters of packets received. There are many different types of controllers available. Most of the controllers are java and Python-based. e.g., Pox [9], NOX [10], OpenDaylight [11], Floodlight [12], RUY [13], etc. SDN has significant advantages over traditional computing paradigms because it reduces capital expenditure (CapEx) and operational expenditure (OpEx) [14]. The integration of SDN and CCE improves cloud availability, manageability, scalability, controllability, and dynamism. Cloud computing provides a networking-as-a-service (NaaS) model using the Network function virtualization (NFV) [15] and SDN provides a specialized platform for NFV [16]. Using SDN, network devices can be programmed according to the requirements and needs, which can efficiently handle EDOS attacks.

EDOS attacks tend to be a crucial concern in cloud computing environment that needs to be tackled. J. Idziorek in [17] did a mathematical calculation to find the cost for routine web requests. Size of 320 KB web traffic is sent with a rate of One Request per Minute for One Month, and 13.18 GB of extra data in one month was observed. Similarly, an experiment was performed in 2011 [18], 1000 requests per second were sent to a web-service hosted on Amazon CloudFront for 30 days with 1000 Mbps data rate. These additional requests cost an extra $42,000 in 30 days. In 2015 most of the attack targets were cloud services [19]. In March 2015, Greatfire.org, on Amazon EC2 cloud, was charged with an enormous bill of $30,000 daily because of massive DDoS attacks [20]. In 2014, the average financial damage by DDoS attack was $444,000 [21].

According to the 13th International worldwide infrastructure security report [22], the most massive reported attack in 2016 was 800 Gbps, which was 8 Gbps in 2004. More than 64% of DDoS attacks in 2017 have targeted more than one vector, and the most prolonged attack lasts 174 h and 53 min [3]. According to Q1 2018 report [23], there is an increase of 53% in a number of DDoS attacks compared to 2017, a 47% increase in the average of attack peak sizes in 2018 compared to 2017, 74% of attacks peaked over 1 Gbps.

These attacks are different in nature and depending on the nature of attack the mitigation techniques can also be different. After studying different techniques, EDOS-IDM is proposed in this paper.

This paper proposes an SDN based computationally cost-efficient technique to mitigate volumetric ICMP DoS/DDoS and Normal behavioral ICMP DoS/DDoS attacks by identifying normal traffic behavior and using the exponential back-off function. To prevent ICMP attack from a single source, the proposed model allows a single flow from a specific source to a specific destination and drops for time "t". If the flow continues even after time "t", the proposed model allows the flow again to prevent legitimate user's flows from starvation by false negatives. The proposed SDN based proactive technique is independent of network topology and traffic pattern and is tested in a real-time cloud environment. The proposed technique is compared with normal behavioral ICMP DDoS attacks because according to our extensive study, existing DoS/DDoS mitigation techniques cannot detect normal behavioral ICMP traffic attacks. The proposed technique takes only "n" seconds to detect and mitigate an attack, which will be set by cloud providers. The "n" seconds is a time in which flow statistics from SDN switch will be forwarded to SDN controller for decision.

The rest of the paper is organized as follow. Section II introduces the schemes to mitigation EDOS and DDoS attacks. In section III, the proposed model, EDOS-IDM is explained. In this section, the algorithm and mathematical model of proposed technique is also presented. In Section IV, Experimental setup along with different parameters used in experiments are discussed. Section V contains Results and Discussion. In this section, the performance evaluation of the proposed technique is depicted and discussed. The section VI is the conclusion.

2. Mitigation of EDos and DDos attacks

Mitigating EDOS attacks is of crucial importance to prevent customers from extra financial damages. Many researchers have contributed to mitigate DDoS attacks that results in EDOS attacks. Defense mechanisms against DDoS can be divided into three main types. Signature-based detection, Anomaly-based detection, and Hybrid detection techniques.

The *signature-based* detection mechanism is used against the known type of attacks, in which incoming traffic pattern is compared with known signatures of attack stored in the knowledge base. Bakshi and Yogesh [24] proposed DDoS detection using SNORT, a Signature-based technique in Cloud. Pedro Manso et al. [25] proposed a SNORT based technique for early detection and mitigation of DDoS attacks using SDN. Lonea et al. [26] deployed VM based IDS using Barnyard tools to capture attacks and SNORT to defend against known attacks. Karnwal et al. [27, 28] proposed a filter tree approach against DDoS attack. The major drawback of signature-based intrusion detection is zero-day attacks are undetectable and IDS must be up to date. Compared to signature-based detection, Anomaly-based detection techniques are very efficient and accurate [29].

Anomaly-based detection techniques compare the incoming traffic pattern with the normal traffic pattern over a predefined period. Many researchers have contributed to Anomaly-based detection techniques, which is further divided into Artificial Intelligence-based [30–34], Classifier based [35–40], Machine Learning based [41–45], Data mining based [46–50] and Statistical anomaly detection techniques [51–64]. Most of the Anomaly-based detection mechanisms are based on Statistical Anomaly detection, which is helpful in terms of volumetric based attacks [45].

The problem with AI-based techniques is scalability and overfitting during the training phase. Classifier based methods require adequate training information to detect unfamiliar outbreaks and resource consumption can be comparatively high. Machine learning-based techniques can suffer from performance degradation because they need significant computational resources for both testing and training phases. Statistical anomaly detection must setup an optimal threshold, and it also involves hypotheses and assumptions, which must be justified reasonably. Data mining-based detection techniques can worsen performance to select attributes from large datasets[2].

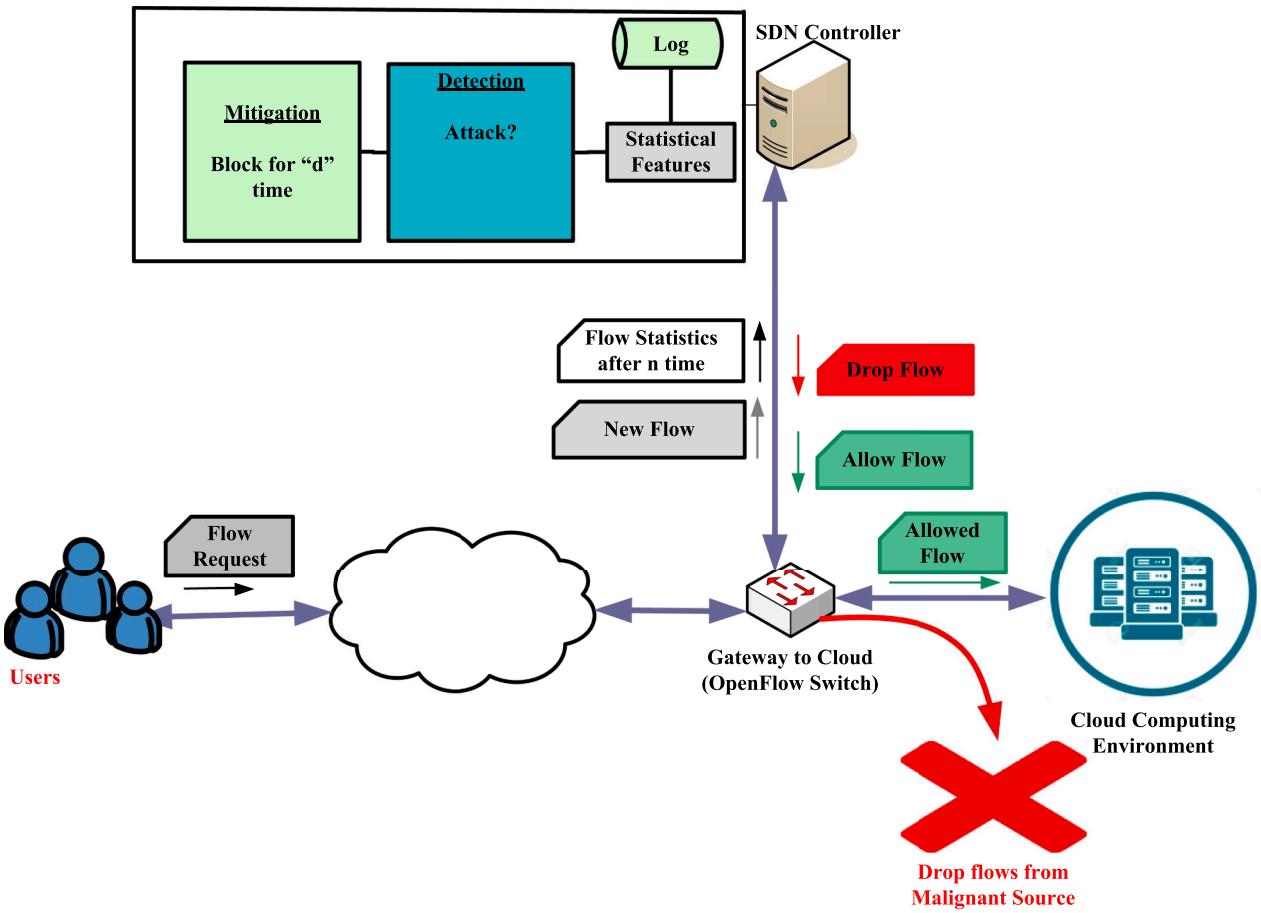

Fig. 1. EDOS-IDM System Model.

Hybrid detection techniques use complementary features of both anomaly-based and signature-based methods to achieve a better detection rate. R. Gopeshwar Rao in [65] proposed a proactive hybrid detection technique SEDoS-7 using SDN in cloud computing to mitigate EDOS attacks. Al-Haideri et al. [66] proposed a hybrid detection technique EDOS-Shield to mitigate EDOS attacks. Similarly, many researchers [67–73] proposed DDoS mitigation techniques but the major drawback of these types of methods is overhead. The comparative summary of all the DDoS mitigation techniques are shown in Table 1.

Peng et al. proposed [74] History-based IP filtering (HIP) mechanism in order to prevent DDoS attacks. Based on history of DDoS attacks, once a history table is maintained, server checks the history for new incoming connection and block or allow the connection based on that history. The problem with HIP is that if attacker knows the technique, attacker can mislead the server by generating a normal connection first to be added in whitelist of the server and then launch ICMP based DDoS attack.

J. Udhayan et al. [75] proposed a rate limiting technique to minimize the impact of ICMP flooding attack on a server. Using this technique, based on link bandwidth, a portion of available bandwidth is allocated as threshold to all ICMP traffic. This is one of the state-of-the-art approaches but has two problems. First is the bandwidth of the link because all links do not have same bandwidth and the link bandwidth must be defined manually to define threshold. Second is, this approach can work for servers to minimize the impact of DoS/DDoS attacks but this solution do not cater EDOS attack, even rate limiting technique has low impact but still resources of server consume because of these attack connections.

CHEN Xiuzhen et al. [76] proposed probabilistic marking approach with multi-tag for tracing ICMP DoS/DDoS attacks. In this approach each router, prior to forwarding the incoming ICMP packet, add its IP address and TTL in data field of the ICMP request packet. When the server receives the ICMP request packet it can trace-back the initiator of the connection and can help the server to detect the source of attack. The problem with this technique is overhead because each router adds IP address and TTL in each ICMP request packet before forwarding. Another problem is to train each router or modify the control process of each router. By adding the IP address and TTL of each router in a path of ICMP request packet, the size of packet increases, and the received bandwidth of a server rises.

M.A. Vinothkumar et al. [77] proposed a technique to block high and low-rate ICMP flooding attacks. In their proposed technique, a combination of Firecol and HAWK are combined to detect high-rate and low-rate attack. Their proposed technique detects high-rate attack using ISPs allocated bandwidth and received bandwidth. If the bandwidth received is greater than ISPs allocated bandwidth, it is concluded as attack. For low-rate ICMP flooding attack, in their paper Halting anomalies with weighted choking (HAWK) is used. HAWK is a technique originally used for TCP flooding attack detection by Yu-Kwong Kwok [78]. HAWK assigns threshold values to all incoming packets. If the received packets are showing large variation, it is sent to Intrusion Prevention system (IPS). On detecting the packet as malicious, the information is propagated to all IPS. The limitations of this technique are IP and port number are blocked but ICMP does not have any port number it uses port zero, a wildcard port.

Harshita [79] proposed a rate limiting technique for ICMP packets. According to this technique, a bandwidth limit is applied for ICMP packets and the threshold value of 1000 bits/sec is used. According to the authors, if any ICMP packet exceeds this value, router discards that packet. If an attacker has the knowledge of the technique used in the router, he can mislead the technique by sending packets size less than

Table 2
Notations of parameters and feature used in algorithm.

Sr #	Feature and Parameters	Notation
1	Source IP	IP_{source}
2	Destination IP	IP_{dest}
3	Protocol	P
4	size of icmp packet received	$\acute{S}$
5	Mean IAT of packets	$E[t]$
6	Number of packets against each flow	η
7	Flow duration	T_{flow}
8	flow identified as icmp	*Icmp_flow*
9	Allowed time of ICMP flow	α
10	Allow function to forward the packets by switch	*allow(time)*
11	Drop function to drop certain flow for certain time	*drop(time)*
12	ICMP Type field value	*icmp_type*
13	Base of exponential back-off function	n
14	Exponent of Exponential Function	x
15	Exponential Back-off function	*backoff(n,x)*
16	Back-off value calculated from backoff(n,x)	$T_{Backoff}$
17	Maximum $T_{Backoff}$	τ
18	Allowed time of Normal behavioral flow	ψ
19	Drop time of flow	δ
20	Flow found in log	φ
21	Add new flow in log	*addflow*(IP_{source},IP_{dest}, $T_{Backoff}$, $Tf_{low,\eta}$)
22	Search function to search for log entry maintained on controller	*search(log)*
23	Update log entries	*update(log)*
24	Get function to get statistics from switch	*getstats(statistics)*

1000 bits/sec. Furthermore, this technique is not useful in CCE because cloud does not suffer from DoS/DDoS attacks, using auto-scaling feature in CCE, it scales resources dynamically and leads to EDOS.

Based on the literature there is a need of such technique that can detect and mitigate Volumetric and Normal Behavioral ICMP traffic attack. Detecting EDOS and DDoS attacks is not an easy task to perform, a lot of contribution is done to detect and mitigate DDoS attacks, but if the attack pattern is similar to regular traffic, it is challenging to detect such attacks and results in false positives, and additionally, a large number of resources are needed for these techniques to mitigate DoS/DDoS attacks.

This paper presents an SDN based proactive "EDOS-IDM" model, that can detect and mitigate both volumetric and normal behavioral ICMP traffic attack. Unlike other anomaly detection approaches, Statistical anomaly detection mechanism is efficient with no overhead, no scalability issue, and no overfitting problem. The only problem with statistical models is the lack of adaptivity. To overcome the issue of lack of adaptivity, adaptive feature of SDN is used with statistical approach. EDOS-IDM uses statistical approach and focuses on different statistical features such as ICMP request packet size, Mean Interarrival time (IAT) of ICMP requests, flow duration and exponential back-off to block attack flows.

3. EDOS- ICMP Flood attack detection and mitigation (EDOS-IDM)

This section explains the proposed ICMP flooding attack Detection model to mitigate EDOS attacks. The model focuses on a proactive approach of SDN in the CCE to mitigate volumetric and normal behavioral ICMP DDoS attack resulting EDOS. One of the general practices to encounter ICMP flooding attacks is to block all the ICMP packets by defining a rule in the firewall. This is not referred to as a significant solution, because ICMP protocol can be used by other protocols for route discovery, shortest path discovery, round trip time (RTT) as well [80]. EDOS-IDM does not block ICMP traffic completely to prevent legitimate user's requests from starvation of false negatives and legitimate users can diagnose and troubleshoot connectivity issues or can use other protocols that used ICMP protocol. The description of the model is given in Section 3.1.

3.1. ICMP attack detection and mitigation using statistical features and exponential back-off

When the first ICMP packet arrives at the OpenFlow switch, it is forwarded to the SDN controller for decision. The controller inspects the packet, and on identification of the ICMP request, the controller first checks the received packet size. In Microsoft Windows, the default PING request is 32 bytes, and in Linux operating system such as Ubuntu, the ping request size is 64 bytes by default. After approval of flow to pass, switch saves the flow statistics and based on the statistics IAT is calculated by controller and detects the flow behavior as normal or volumetric. After IAT controller inspects the flow duration and based on that duration controller decides to continue the flow or use exponential back-off function to drop flow for that time. The legitimacy of the flow sill be calculated using the parameters discussed. The EDOS-IDM system model is shown in Fig. 1 and the detailed working of the model is shown in Table 6.

The features used for this research study are "IP_{Source}, IP_{dest}, Protocol P, size of packets $\acute{S}$, mean interarrival time (IAT) of packets E[t], Number of Packets against each flow η, Time of flow T_{flow} and back-off value $T_{Backoff}$". The details of features and parameters used in Table 6, are given in Table 2.

For exponential back-off function, the previous state of back-off value is saved against each flow in the log file and new back-off time for flow is calculated, if the flow needs to be dropped. If the flow is already present in the log then the controller does not add, Source and Destination IPs in the log, and parameters related to the flow are updated only.

The mechanism for detection and mitigation of ICMP attack in the EDOS-IDM model with exponential back-off is shown in Table 6 and according to Asymptotic Analysis, the complexity of the algorithm is O (n).

On receiving the first packet by the controller, the controller checks the protocol in the protocol field of the IPv4 header. On recognizing the protocol as ICMP, checks for the packet size in step-5 of Table 6. The default size of the ICMP packet is 64 bytes or less. If the controller detects that the packet size is greater than 64 bytes, the controller straight forwardly drops the packet and instructs the switch to block this specific flow. If the condition is not satisfied, checks the flowtime of the flow, if the flowtime is less than α and the packet received is first packet of the flow, so it is not in the log, The flow added to log and is allowed to pass for α time. After hard timeout or soft timeout, all the flow entries in switch flush and after flushing the entries, the switch sends the new received packet to controller for decision. This time controller checks the flow in the log and if it is present in the log, it checks the flow duration and blocks the flow for the exponential time $T_{Backoff}$ value, calculated by backoff function, if the flow duration is greater than α. The $T_{Backoff}$ time increases exponentially till threshold τ and update in the flow log on every iteration and updates the flow statistics as well. After reaching the $T_{Backoff}$ to threshold then divide it by two for next iteration to keep false negatives in mind. In the next step the controller checks for flooding attacks by calculating mean IAT *E[t]* of ICMP request packets in step-43. If the value of *E[t]* is greater than 1 this means that number of requests in a given time is greater than normal expected, it is considered as flooding. By default, a system sends ICMP requests after 0.8 s to 1 second under normal condition. If flooding is observed, block the flow for δ time. The mean IAT of ICMP request in a flow can be calculated using Equation-1.

$$E[t] = \frac{\eta}{T_{flow}} \tag{1}$$

In Equation-1 "η" stands for ICMP requests and "T_{flow}" stands for an active time of flow. The drop time δ is calculated using exponential back-off function "*backoff(n,x)*". If the $T_{Backoff}$ value that is assigned to δ, reaches threshold τ as shown in step-20, then divide the back-off by 2 to prevent the back-off from reaching infinity to control the system's false negatives. In normal behavioral ICMP attacks, normal ICMP requests continue to be sent from multiple systems and the requests may continue for days or even months. Step-50 shows the mitigation of such types of attacks, where T_{flow} is compared with threshold ψ and on the success of the condition, blocks the flow for $T_{Backoff}$ and allows the flow after the $T_{Backoff}$. Here one point is worth noting that this should be properly communicated to the customer in service level agreement (SLA), and the cloud providers must communicate the values of α and δ as per their requirement.

The mathematical model of the system is shown in eq-2, which shows total resource usage "ρ" of an instance and based on this usage client is charged.

$$\rho = \sum_{i=1}^{t} \sum_{h=1}^{k} \left(\theta_{h,i} \times R_{h,i}\right) \tag{2}$$

Where θ is calculated using eq-3

$$\theta = \begin{cases} 1 & \begin{cases} < \text{ct} > \acute{S} < \text{ot} > \leq 64B \\ E[t] \leq \tau \\ T_{flow} \leq \psi \end{cases} \\ 0 & Otherwise \end{cases} \tag{3}$$

In equation-2 "i" is running time of controller, "h" is the number of systems from which ICMP attack or normal ICMP flow is initiated. "$R_{h,i}$" represents the resource usage of an instance, at the time "i" by host "h". "R" can be any resource such as bandwidth, CPU utilization etc. If an attack is initiated from multiple hosts, the resource usage of an instance by all hosts "h" at the time "i" is added. The resources will only be added, if the value of "θ" for a host "h" at the time "i" is one. The multiplicative Identity property of one considers the resource usage and if the value of "θ" is zero, upon multiplication with zero, resources is not considered as used and client is not charged for that.

Equation-3 calculates the value of "θ", which is a binary value. If the value of "θ" is one, this shows resource consumption and zero means no resource utilization. The resource is consumed if the conditions are satisfied. The first condition is "$\acute{S} \leq 64B$", "$\acute{S}$" stands for packet size. If packet size of ICMP request is 64 bytes or less, the controller allows the flow to forward and value of "θ" is one, but in case of ICMP request packet size greater than 64 bytes, the controller drops the flows and value of "θ" is set to zero. If the first condition satisfies then controller inspects second condition. The second condition, *"$E[t] \geq \tau$"* shows ICMP request IAT. In the experiments performed, the *"τ"* value is set to 0.8 s. Initially, the controller has no flow information, in the experiments, the controller allowed the flows for four seconds. After four seconds controller collects statistics from the switch and calculates *E[t]* according to equation-1. If the second condition satisfies, the value of "θ" is set to one and zero otherwise. On failure of the condition, the controller blocks the flow for δ time, which is an exponential back-off value and in the experiments performed, the exponential back-off function used in 5^x, where value of n is set to 5 and the value of "x" is set to 1 on initial back-off i.e. flow is dropped for 5 s. After the expiry of the back-off timer, the controller again allows the flow for 4 s and after four seconds on the failure of the second condition, the controller increments *"x"* by one and value of *"x"* is set to two. So, the controller drops the flow for 25 s. If both of condition-1 and condition-2 satisfies, third condition i.e. "$T_{flow} \leq \psi$" is checked. This condition is for the detection and mitigation of normal behavioral ICMP traffic attacks. In the "T_{flow}" shows flow duration and "ψ" is the threshold value. On the success of the first two conditions, the controller allows the flows to forward, but if the flow continues, it can lead to a normal behavioral ICMP traffic attack. In the experiments performed in the paper, "ψ" is set to 25 s. After 25 s of flow, the flow is dropped for δ time which is an exponential back-off value and in the experiments performed in the paper, the exponential back-off function used is 5^x and works in the same fashion as in condition two.

Table 3
Nodes required for OpenStack production cloud.

Sr. no.	Node	Number of machines
1.	Metal as a Service (MaaS)	1
2.	Juju node	1
3.	Compute Nodes	3
4.	Network Management Node	1

Table 4
Monitored parameters.

Sr. No.	Parameter	Description
1.	Softirq(Rx)	Softirq is software interrupts and is a very important and critical matric for OS Kernel. The Softirq mechanism is meant to handle processing and run at a high priority
2.	Softnet	Softnet is processed events related to network traffic
3.	ICMP packets received	The number of ICMP packet received by an instance
4.	Bandwidth (Rx)	Consumed bandwidth by inbound traffic

4. Experimental setup

The experimental setup used for the research is the OpenStack Production Cloud Environment. The cloud environment composed of six Machines shown in Table 3, each containing two Network Interface Cards. One NIC is for connectivity of the machine to the Internet, and another NIC is used for connectivity within Cloud Environment.

Here one point is worth mentioning that the network management node manages only the Cloud virtual network. The cloud environment is connected to the Internet. On the gateway of Cloud Environment, the SDN switch is used for traffic monitoring by SDN controller. All the requests from the Internet to the cloud environment must pass through the SDN switch. SDN switch is a dummy switch because it has only a data plane. The controller decides action and the switch acts accordingly. An instance is created in a cloud environment that supports autoscaling, such that resource utilization is scaled automatically as per-use and launches attacks from different sources on this instance.

4.1. Traffic generation

For ICMP attack traffic, "HPING3" application is used and for normal traffic "ping" application is used. To test the model for ICMP flooding attacks and normal behavioral ICMP traffic attack, the packet sizes used are 65 KB and 64 bytes with interarrival time of one second for normal behavioral ICMP traffic and set 0.2 s for ICMP flooding for the proof of experiments but the model can work for any packet size and any packet IAT.

4.2. Parameters monitored

To observe the behavior of EDOS-IDM model, the parameters monitored in experiments are shown in Table 4.

CCE uses OpenVswitch for management and internal communication of the cloud. Using the EDOS-IDM model in the SDN controller with OpenVswitch, CCE will also be safe from inside ICMP flooding threats. For real-time network monitoring, the netdata tool is used. The netdata tool monitors CPU, Memory, Network, etc. The results are accumulated from the netdata tool and plotted in MATLAB 19b.

In all the experiments, launch attacks from 20 machines on a single instance and monitor the resources of targeted instance. The details of experiments are given as under,

Table 5
Network parameters of Experiments performed.

Experiment no.	Attack packet size	IAT	Attack type
1.	65KB	0.2 s	Flood
2.	65KB	1 s	Normal Behavioral
3.	64B	0.2 s	Flood
4.	64B	1 s	Normal Behavioral

4.3. First experiment (Large packet/small IAT)

In the first experiment, an ICMP attack is launched from 20 hosts on a single instance of OpenStack cloud. The ICMP packet size 65 KB and 0.2 s static inter arrival time is set among ICMP requests, for the proof of experiment. According to 0.2 s IAT, 20 hosts send 100 packets per second, whereas normal ping request has approximately 1-second mean inter arrival time between packets and total 20 packets per second is sent from 20 hosts. The instance responds with a ping reply against each ping request with 65 KB payload. The results are accumulated for both EDOS-IDM and without EDOS-IDM

4.4. Second experiment (Large packet/default IAT)

To verify the performance of the system another experiment is performed with ICMP request with packet size 65 KB but the IAT between requests was set to default instead of 0.2 s. The attacks on instance are launched from 20 hosts.

4.5. Third experiment (Default packet/small IAT)

After evaluating the performance of the proposed model against the 65 KB attack, in a series of experiments, the model is also tested for 64 bytes attack against ICMP flooding with a time interval of 0.2 s. The attack is launched from 20 hosts on a single instance of OpenStack production cloud.

4.6. Fourth experiment (Default packet/default IAT)

After evaluating the performance of EDOS-IDM model against Flooding attack and ICMP packet size attack, where payload is added in ICMP packet to consume bandwidth, a last experiment is performed. In this experiment, EDOS-IDM model is tested for normal behavioral ICMP

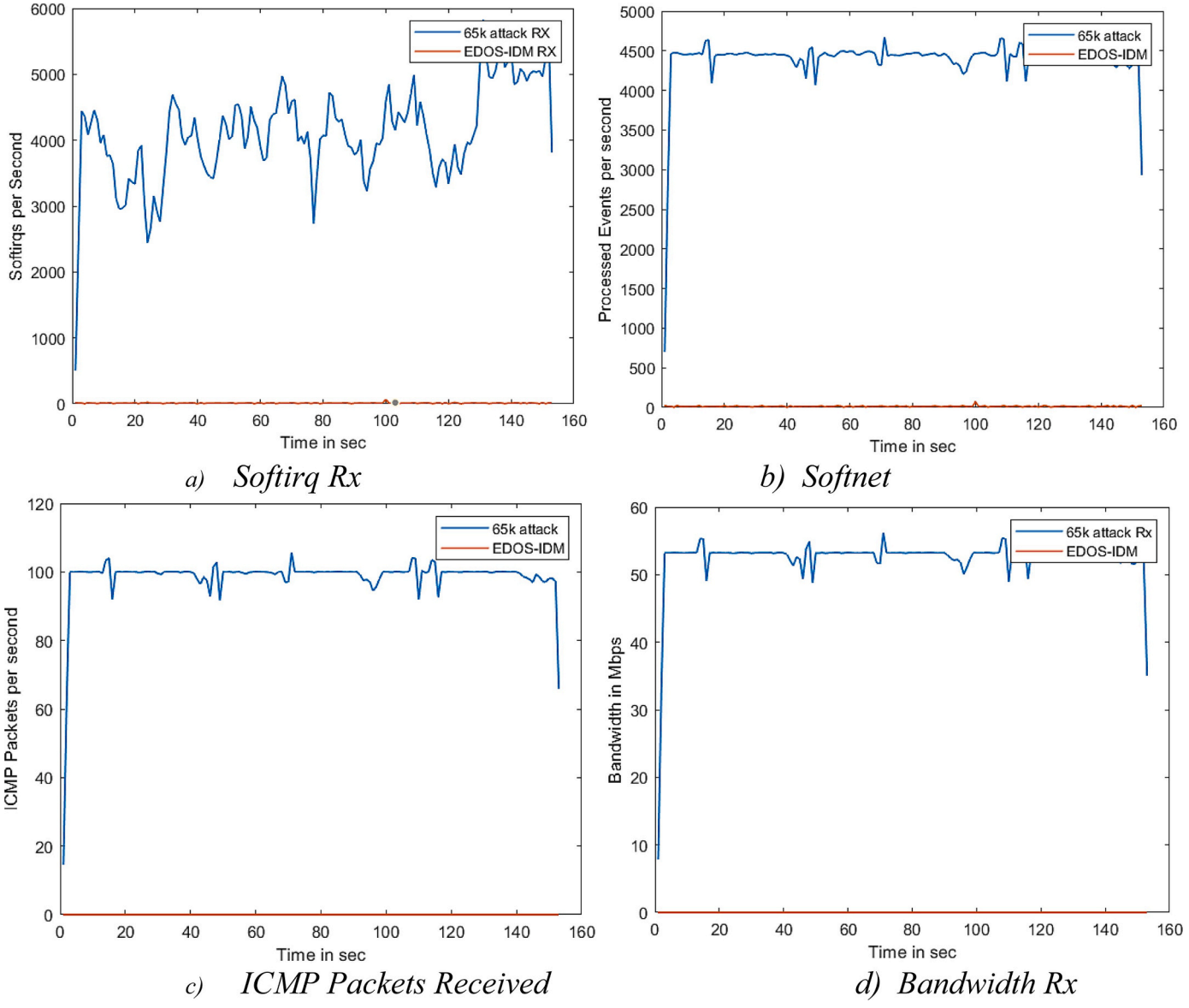

Fig. 2. 65 K Attack vs EDOS-IDM with ICMP request interval 0.2 S.

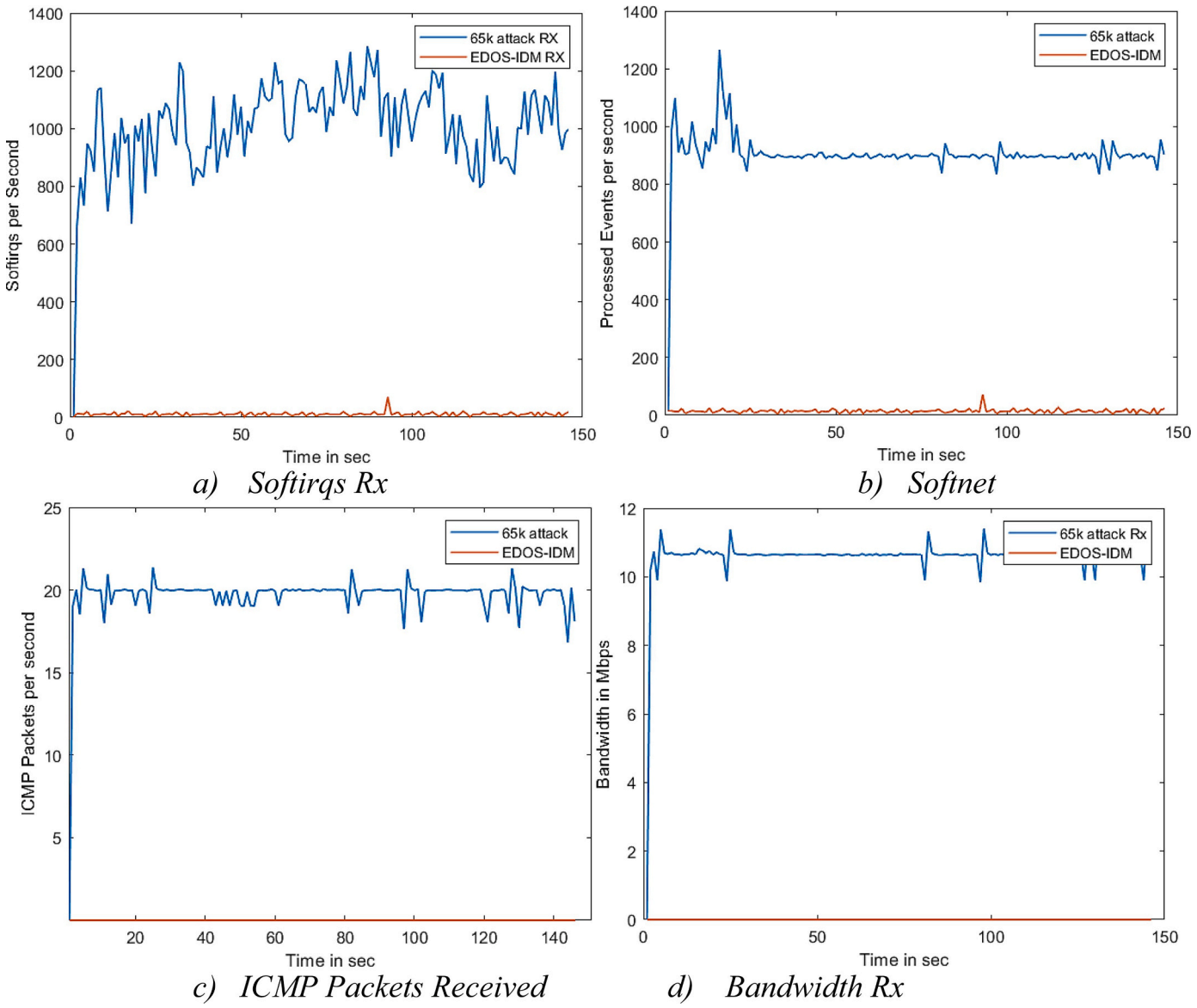

Fig. 3. Attack with packet size 65 KB vs. EDOS-IDM with ICMP time interval 1 second.

traffic attack, the most important attack to be detected. In this experiment, the resource usage of instance is observed for EDOS-IDM and without EDOS-IDM, against normal behavioral ICMP traffic attack.

5. Results and discussion

ICMP flooding and normal behavioral ICMP traffic attacks from hosts outside the cloud environment are launched, on an instance running in OpenStack Cloud Environment to measure the influence of ICMP flooding attack. The traffic and system resources are observed by the netdata tool for both EDOS-IDM mode and Normal behavioral ICMP traffic attack. The network parameters used in performed experiments are shown in Table 5.

The results achieved from experiment 1 is shown in Fig. 2.

Fig. 2a shows softirqs experienced due to data received on the network interface. In this figure, at the time of 155 s, the attack was stopped, and the fall can be noticed. The average Softirqs observed during the 65 K attack was 4000 Softirqs per second, whereas zero Softirqs per second was experienced during the EDOS-IDM model because the packet size was greater than 64 bytes. Fig. 2b shows softnet observed due to network traffic, the softnet during 65 KB attack shows an average of 4500 processed events per second and this value also remains zero for EDOS-IDM. Fig. 2c shows ICMP packets received during the 65 KB attack and EDOS-IDM. During the 65 KB attack, an average of 100 ICMP packets per second were observed whereas controller blocked 65 KB ICMP packets in case of EDOS-IDM and zero ICMP attack packets were noticed during EDOS-IDM. Fig. 2d displays the received network bandwidth, i.e., 65 KB attack consumed network bandwidth is 54 Mbps on average, on the other hand, EDOS-IDM consumes no bandwidth. With similar fashion, if the attack continues for a month the resource usage by 65 KB attack can be calculated using formula $r \times 60 \times 60 \times 24 \times 30$, where r is average resource usage per second.

The results collected from Experiment 2 are shown in Fig. 3. Fig. 3a shows Softirqs observed due to the received network traffic. In this figure, an average of 1000 Softirqs per second can be noticed whereas EDOS-IDM shows an average of 10 Softirqs per second but the Softirqs observed during EDOS-IDM were not because of ICMP packets because zero ICMP packets can be observed during EDOS-IDM in Fig. 3c. The Softirqs experienced during EDOS-IDM is because of network traffic other than ICMP traffic. Fig. 3b shows the softnet i.e. processed event per second. In the figure, 65 KB attack shows an average of 950 events processed per seconds and EDOS-IDM shows about zero processed

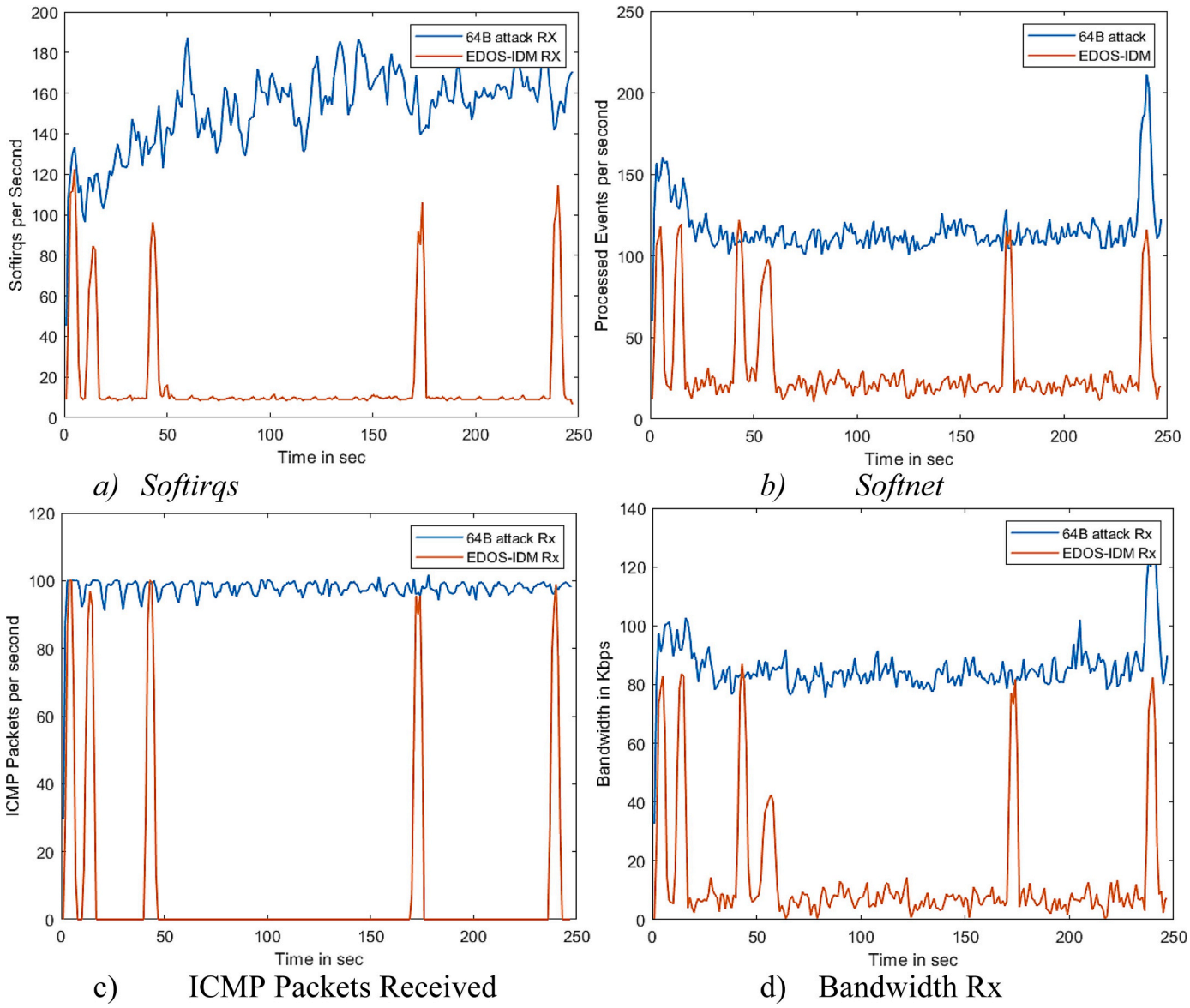

Fig. 4. Attack 64 bytes vs EDOS-IDM with Time Interval 0.2 s.

events per second. The ICMP packets received by an instance shown in Fig. 3c. In the figure, 65 KB attack shows an average of 20 ICMP packets per second. Whereas after using EDOS-IDM, SDN controller blocks all ICMP packets of 65 KB because controller recognizes ICMP packets greater than 64 bytes as an attack. So, no ICMP attack reaches intended instance. The received bandwidth results are shown in Fig. 3d. In this figure 65 KB attack shows about 11 Mbps received network traffic whereas, EDOS-IDM shows zero Mbps received traffic because no ICMP packet of size 65 KB was allowed by the controller to be forwarded.

Fig. 4 shows the results achieved from Experiment 3. Fig. 4a shows the Softirqs observed during the 64 bytes attack and EDOS-IDM. Using 64 bytes attack there is no mechanism to detect ICMP flooding attack and all the ICMP requests are allowed to forward. The average Softirqs received during the 64 bytes attack are 160 Softirqs per second whereas, during the EDOS-IDM model, the Softirqs observed are 12 Softirqs per second. Using EDOS-IDM, Controller verifies the packet size and on verifying the packet size equal to or less than 64 bytes, the controller allows the flow from a source to a destination and saves the flow in the log file. EDOS -IDM mitigates ICMP attack based on packet size as well as ICMP requests' IAT. The exponential function for this experiment is "5^x" and is shown in steps 16 to 18 of Table 6. After allowing the flows to be forwarded, the controller gets statistics from the switch after four seconds. The controller observes the ICMP requests are greater than normal and IAT among ICMP requests were less than normal in four seconds, so the controller drops the flow for 5^1=5 s. After 5 s controller allows the flow for four seconds again to prevent the user requests from false negative. After four seconds, the controller gets the statistics and updates the log file. The controller observes the flooding from source to the destination once again and this time blocks the flows for $5^2 = 25$ s. After 25 s, the controller allows the flow to forward for 4 s and then blocked for 5^3=125 s and the process continues in the same fashion. When the exponential back-off value reaches threshold value i.e. 125 was set in this case, after allowing the flows for four seconds the controller blocks the flows for threshold $\div\ 2 = x$ seconds. The rise in Fig. 4 can be observed at 1 to 4 s, 9 to 13 s, 38 to 42 s, 167 to 171 s and 233 to 237 s of EDOS-IDM, where traffic is allowed to be forwarded by controller. The average softnet for 64 bytes attack is 110 processed events per second whereas for EDOS-IDM, the softnet noted is 28 processed events per second shown in Fig. 4b. The number of ICMP packets is shown in Fig. 4c. In the figure, 64 bytes attack shows 100 ICMP requests per second, but the no noticed ICMP requests during the EDOS-IDM model is Eight ICMP packets per second. The received bandwidth detected during

Algorithm 1:
ICMP Detection and Mitigation Model with Exponential Back-Off time.

```
Start
DATA: icmp_flow, Ś, α, icmp_type, IP_source, IP_dest, T_back-off, n, τ, δ, φ, T_Backoff,x
OUTPUT: Flow is granted admission or dropped
if flow ⟵ icmp_flow then
    if Ś > 64 bytes then
        block the flow
    else
        if T_flow <=α then
            IP_source=source_ip
            IP_dest=destination_ip
            φ=0
            if icmp_type=8
                search(log)
                for each flow f in the log, compare IP_source and IP_dest
                    if IP_source and IP_dest flow, found in log
                        backoff (n,x)
                          T_Backoff =n^x
                          return T_Backoff
                        φ=1
                        if T_Backoff > τ
                            T_Backoff = τ ÷ n
                        end if
                        δ = T_Backoff
                        break
                    end if
                end for
            end if
            if φ=0 and icmp_type=8
                T_Backoff = 1
                T_flow=0
                addflow(IP_source,IP_dest, T_Backoff, T_flow,η)
                δ = T_Backoff
                allow(α)
            else
                if φ=1then
                    backoff (n,x)
                        T_Backoff =n^x
                        return T_Backoff
                    update(log)        //update value of β of flow present in log
                    getstats(statistics)
                    update(log)
                    E[t] = η ÷ T_flow
                    if E[t]>1 then
                        if T_Backoff > τ then
                            T_Backoff = T_Backoff ÷ 2
                            δ = T_Backoff
                        end if
                        drop(δ)
                    else
                        if T_flow > ψ then
                            if T_Backoff > τ then
                                T_Backoff = T_Backoff ÷ 2
                                δ = T_Backoff
                            end if
                            drop(δ)
                        else
                            allow(ψ)
                        end if
                    end if
                end if
            end if
            allow(α)
        else
            drop(δ)
        end if
else
    Allow other types of flows
end if
Finish.
```

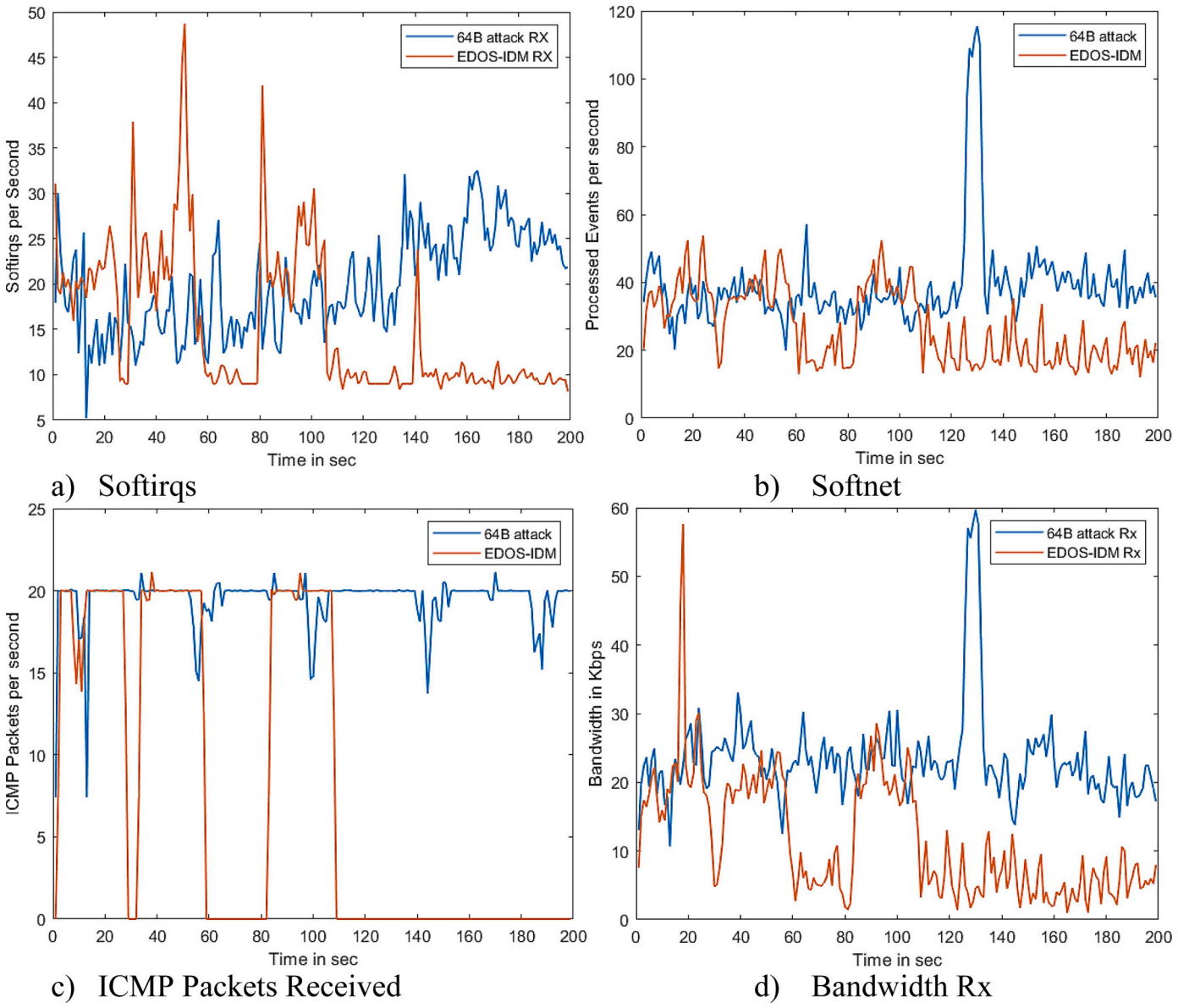

Fig. 5. 64 bytes vs. EDOS-IDM 1 s interval.

64 bytes attack is 85 Kbps but EDOS-IDM received bandwidth is 13.5 Kbps.

The results achieved from Experiment 4 is shown in Fig. 5, it shows normal behavioral ICMP traffic and compared with the CCE using EDOS-IDM with exponential back-off function 5^x. Unlike flooding attack, where the controller was allowing the flow for four seconds, normal behavioral flow is allowed for 25seconds to prevent the system from false negatives. Fig. 5a shows 20 Softirq per second during normal behavioral ICMP traffic attack and 15 Softirqs per second while using EDOS-IDM. In Fig. 5b softnet shows 38.9 processed events per second for normal behavioral ICMP traffic attack and 26.5 processed events per second in the case of EDOS-IDM. The number of ICMP received packets by an instance is shown in Fig. 5c. The observed ICMP packets during normal behavioral ICMP traffic attacks are 19 and only seven ICMP packets seen against EDOS-IDM. The received network bandwidth in Fig. 5d for normal behavioral ICMP traffic attack is 23.4 Kbps. If the attack continues for a month, the client is charged against 7.5GB of inbound traffic and 7.5GB of outbound traffic. EDOS-IDM, in this case, can perform much better. The client will observe only 11.13 Kbps received bandwidth and the received bandwidth in a month will be 3.6GB and the outbound network traffic will also be 3.6GB in a month.

The results show that, whatever the IAT is, EDOS-IDM model shows zero softirqs, softnets, ICMP packets received and bandwidth received, if a payload is added in ICMP request packet and packet size exceeds 64B as in Experiment 1 and Experiment 2. In case of the packet size less than or equal to 64 bytes, the model then checks for IAT primarily. If the IAT is less than default as shown in Experiment 3, the controller allows the packets for four seconds only. On receiving the statistics by controller from SDN switch, the controller detects this as a flooding attack. It blocks the flow exponentially and reduces the resource consumption as shown in the experiment. In case of default IAT and packet size less than or equal to 64B, the model detects the traffic as normal ICMP traffic. The model allows the traffic for 25 s as shown in experiment 4. But if the flow continues even after 25 s, it blocks the flow according to exponential function. When the drop time expires which is 5 s in the experiment, the controller checks that the flow is still active, controller allows the flow for 25 s again and after 25 s blocks the flow for 25 s through exponential function of 5^x and the process continues in the similar fashion.

From the discussion, the EDOS-IDM model performs better than other DDoS mitigation techniques in the CCE because the EDOS-IDM

model can mitigate volumetric as well as normal behavioral ICMP traffic attacks with less computational cost. EDOS-IDM takes a fixed time "n", which will be set by cloud providers in the SDN controller for getting statistics from the switch, to block both Volumetric and Normal Behavioral ICMP Traffic attack.

6. Conclusion

Resources in CCE scale dynamically according to requirements. Intrusions in CCE play a vital role in financial damages to customers because of enormous resource needs and thus lead to an EDOS attack. The most common and easy to launch a DoS/DDoS attack is the ICMP flooding attack. The cloud instances are accessible online through network, so, the core medium of attack is network. Securing the Network results in cloud security from DoS/DDoS attacks. Using traditional network equipment, it is not possible to tackle intrusion. This research focuses on the use of a Software-Defined network, an open architecture for the network, and program network devices according to network requirements. This research proposes the ICMP Detection and Mitigation Model to alleviate the impact of volumetric and normal behavioral ICMP traffic attack on Cloud Environment using SDN. EDOS-IDM model uses an n-time based scheme. In this scheme, ICMP traffic is identified and is allowed for n-time only if the ICMP packet size is less than or equal to 64 bytes. After n-time, the ICMP traffic is blocked by a cloud gateway for d-time. The results prove that a handsome amount of network resources are saved and prevents the customers from paying enormous extra bills because of volumetric and normal behavioral ICMP traffic attacks. In the future, the target of the research is TCP SYN Flooding and UDP Flooding attacks.

Declaration of Competing Interest

None.

Authors statement

We confirm that this work is original and has not been published elsewhere nor is it currently under consideration for publication elsewhere. All authors approved the manuscript and this submission.

References

[1] A. Mahesh, N. Suresh, M. Gupta, R. Sharman, Cloud risk resilience: investigation of audit practices and technology advances-a technical report, Int. J. Risk Contingency Manag. (IJRCM) 8 (2019) 66–92.

[2] O. Osanaiye, K.-K.R. Choo, M. Dlodlo, Distributed denial of service (DDoS) resilience in cloud: review and conceptual cloud DDoS mitigation framework, J. Netw. Comput. Appl. 67 (2016) 147–165.

[3] Nexusguard, DDoS Attacks Now Last Longer And Have Become More Complex, 2018. http://blog.nexusguard.com/ddos-attacks-now-last-longer-and-have-become-more-complex. February.

[4] S. Iqbal, M.L.M. Kiah, B. Dhaghighi, M. Hussain, S. Khan, M.K. Khan, On cloud security attacks: a taxonomy and intrusion detection and prevention as a service, J. Netw. Comput. Appl. 74 (2016) 98–120.

[5] A. Fazli, A. Sayedi, J.D. Shulman, The effects of autoscaling in cloud computing, Manag. Sci. 64 (2018) 5149–5163.

[6] K. Bhushan, B.B. Gupta, Network flow analysis for detection and mitigation of Fraudulent Resource Consumption (FRC) attacks in multimedia cloud computing, Multimed. Tools Appl. 78 (2019) 4267–4298.

[7] R.F. Moyano, D.F. Cambronero, L.B. Triana, A user-centric SDN management architecture for NFV-based residential networks, Comput. Standards Interfaces 54 (2017) 279–292.

[8] C. Cascone, D. Sanvito, L. Pollini, A. Capone, B. Sanso, Fast failure detection and recovery in SDN with stateful data plane, Int. J. Network Manag. 27 (2017) e1957.

[9] S. Kaur, J. Singh, N.S. Ghumman, Network programmability using POX controllerICCCS International Conference on Communication, Computing & Systems, IEEE (2014).

[10] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, NOX: towards an operating system for networks, ACM SIGCOMM Comput. Commun. Rev. 38 (2008) 105–110.

[11] J. Medved, R. Varga, A. Tkacik, K. Gray, Opendaylight: towards a model-driven sdn controller architecture, in: Proceeding of IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks, 2014, pp. 1–6. *2014*.

[12] V.B. Harkal, A. Deshmukh, Software defined networking with floodlight controller, Int. J. Comput. Appl. 975 (2016) 8887.

[13] A. Shalimov, D. Zuikov, D. Zimarina, V. Pashkov, R. Smeliansky, Advanced study of SDN/OpenFlow controllers, in: Proceedings of the 9th central & eastern European software engineering conference in Russia, 2013, p. 1, p.

[14] Z. Xiao, Y. Xiao, Security and Privacy in Cloud Computing," Presented At the IEEE Commun, Surveys Tuts., 2nd Quart, 2013.

[15] M.P.V. Manthena, N.L. van Adrichem, C. van den Broek, F. Kuipers, An SDN-based Architecture for Network-as-a-Service, in: Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft), 2015, pp. 1–5.

[16] M.S. Bonfim, K.L. Dias, S.F. Fernandes, Integrated NFV/SDN architectures: a systematic literature review, ACM Comput. Surv. (CSUR) 51 (2019) 114.

[17] J. Idziorek, M. Tannian, Exploiting cloud utility models for profit and ruin In Cloud Computing (CLOUD), in: presented at the IEEE International Conference on Cloud Computing, 2011.

[18] "ReviewMyLive.co.uk, Amazon CloudFront and S3 Maximum Cost, (http://www.reviewmylife.co.uk/blog/2011/05/19/amazon-cloudfront-and-s3-maximum-cost/)," 2011.

[19] T. Seals, "Q1 2015 DDoS Attacks Spike, targeting cloud, (http://www.infosecurity-magazine.com/news/q1-2015-ddos-attacks-spike/)," 2015.

[20] L. Munson, "Greatfire.org Faces Daily $30,000 Bill from DDoS Attack, (https://nakedsecurity.Sophos.com/2015/03/20/greatfire-org-faces-daily-/30000-bill-from-ddos-attack/)," 2015.

[21] K. Labs, "Global IT Security Risks Survey 2014 – Distributed Denial of Service (DDoS) Attacks, (http://media.Kaspersky.com/en/B2B-International-2014-survey-DDoS-Summery-Report.pdf)," 2014.

[22] Arbor, 13th Annual Worldwide Infrastructure Security Report, October 2017.

[23] Verisign, "Q1 2018 DDoS Attack Trend Report, (https://www.verisign.com/assets/infographic-ddos-trends-Q12018.pdf)," 2018.

[24] A. Bakshi, Y.B. Dujodwala, Securing cloud from ddos attacks using intrusion detection system in virtual machine, in: in 2010 S International Conference on Communication Software and Networks, 2010, pp. 260–264.

[25] P. Manso, J. Moura, C. Serrão, SDN-based intrusion detection system for early detection and mitigation of DDoS attacks, Information 10 (2019) 106.

[26] A.M. Lonea, D.E. Popescu, O. Prostean, H. Tianfield, Evaluation of experiments on detecting distributed denial of service (DDoS) attacks in Eucalyptus private cloud. Soft Computing Applications, Springer, 2013, pp. 367–379.

[27] T. Karnwal, T. Sivakumar, G. Aghila, A comber approach to protect cloud computing against XML DDoS and HTTP DDoS attack2012 IEEE Students' Conference on Electrical, Electron. Comput. Sci. (2012) 1–5.

[28] T. Karnwal, S. Thandapanii, A. Gnanasekaran, A filter tree approach to protect cloud computing against xml ddos and http ddos attack. Intelligent Informatics, Springer, 2013, pp. 459–469.

[29] D. Kwon, H. Kim, J. Kim, S.C. Suh, I. Kim, K.J. Kim, A survey of deep learning-based network anomaly detection, Cluster Comput. (2017) 1–13.

[30] X. Yu, D. Han, Z. Du, Q. Tian, G. Yin, Design of DDoS attack detection system based on intelligent bee colony algorithm, Int. J. Comput. Sci. Eng. 19 (2019) 223–232.

[31] B. Joshi, A.S. Vijayan, B.K. Joshi, Securing cloud computing environment against DDoS attacks, in: 2012 International Conference on Computer Communication and Informatics, 2012, pp. 1–5.

[32] N. Jeyanthi, N.C.S. Iyengar, P.M. Kumar, A. Kannammal, An enhanced entropy approach to detect and prevent DDoS in cloud environment, Int. J. Commun. Netw. Inf. Secur. 5 (2013) 110.

[33] N. Jeyanthi, U. Barde, M. Sravani, V. Tiwari, N.C.S.N. Iyengar, Detection of distributed denial of service attacks in cloud computing by identifying spoofed IP, Int. J. Commun. Netw. Distrib. Syst. 11 (2013) 262–279.

[34] V.S.-M. Huang, R. Huang, M. Chiang, A DDoS mitigation system with multi-stage detection and text-based turing testing in cloud computing, in: 2013 27th International Conference on Advanced Information Networking and Applications Workshops, 2013, pp. 655–662.

[35] A. Chonka, J. Abawajy, Detecting and mitigating HX-DoS attacks against cloud web services, in: 2012 15th International Conference on Network-Based Information Systems, 2012, pp. 429–434.

[36] A. Chonka, Y. Xiang, W. Zhou, A. Bonti, Cloud security defence to protect cloud computing against HTTP-DoS and XML-DoS attacks, J. Netw. Comput. Appl. 34 (2011) 1097–1107.

[37] A.M. Lonea, D.E. Popescu, H. Tianfield, Detecting DDoS attacks in cloud computing environment, Int. J. Comput. Commun. Control 8 (2013) 70–78.

[38] N.C.S.N. Iyengar, G. Ganapathy, P. Mogan Kumar, A. Abraham, A multilevel thrust filtration defending mechanism against DDoS attacks in cloud computing environment, Int. J. Grid Util. Comput. 5 (2014) 236–248.

[39] N. Jeyanthi, N.C.S. Iyengar, Escape-on-sight: an efficient and scalable mechanism for escaping ddos attacks in cloud computing environment, Cybern. Inf. Technol. 13 (2013) 46–60.

[40] R.A. Michelin, A.F. Zorzo, C.A. De Rose, Mitigating dos to authenticated cloud rest apis, in: The 9th International Conference for Internet Technology and Secured Transactions (ICITST-2014), 2014, pp. 106–111.

[41] O. Rahman, M.A.G. Quraishi, C.-H. Lung, DDoS Attacks Detection and Mitigation in SDN Using Machine Learning, in: , 2019, pp. 184–189.

[42] J. Pei, Y. Chen, W. Ji, A DDoS attack detection method based on machine learning, J. Phys. Conf. Ser. (2019), 032040.

[43] S. Gupta, P. Kumar, A. Abraham, A profile based network intrusion detection and prevention system for securing cloud environment, Int. J. Distrib. Sens. Netw. 9 (2013), 364575.

[44] F. Palmieri, U. Fiore, A. Castiglione, A distributed approach to network anomaly detection based on independent component analysis, Concurrency Comput. 26 (2014) 1113–1129.
[45] N. Dayal, S. Srivastava, Leveraging SDN for early detection and mitigation of DDoS attacks, in: International Conference on Communication Systems and Networks, 2018, pp. 52–75.
[46] J. Choi, C. Choi, B. Ko, P. Kim, A method of DDoS attack detection using HTTP packet pattern and rule engine in cloud computing environment, Soft Comput. 18 (2014) 1697–1703.
[47] J. Choi, C. Choi, B. Ko, D. Choi, P. Kim, Detecting web based DDoS attack using MapReduce operations in cloud computing environment, J. Internet Serv. Inf. Secur. 3 (2013) 28–37.
[48] S. Alqahtani, R.F. Gamble, DDoS attacks in service clouds, in: 2015 48th Hawaii International Conference on System Sciences, 2015, pp. 5331–5340.
[49] H. Kwon, T. Kim, S.J. Yu, H.K. Kim, Self-similarity based lightweight intrusion detection method for cloud computing, in: Asian Conference on Intelligent Information and Database Systems, 2011, pp. 353–362.
[50] Z. Chen, G. Xu, V. Mahalingam, L. Ge, J. Nguyen, W. Yu, A cloud computing based network monitoring and threat detection system for critical infrastructures, Big Data Res. 3 (2016) 10–23.
[51] X. Xie, J. Li, X. Hu, H. Jin, H. Chen, X. Ma, High performance DDoS attack detection system based on distribution statistics, in: IFIP International Conference on Network and Parallel Computing, 2019, pp. 132–142.
[52] B.K. Devi, T. Subbulakshmi, Cloud-based DDoS attack detection and defence system using statistical approach, Int. J. Inf. Comput. Secur. 11 (2019) 447–475.
[53] S. Chapade, K. Pandey, D. Bhade, Securing cloud servers against flooding based DDoS attacks, in: 2013 International Conference on Communication Systems and Network Technologies, 2013, pp. 524–528.
[54] A. Shawahna, M. Abu-Amara, A. Mahmoud, Y.E. Osais, EDoS-ADS: an enhanced mitigation technique against economic denial of sustainability (EDoS) Attacks, IEEE Trans. Cloud Comput. (2018).
[55] P.S. Bawa, S.U. Rehman, S. Manickam, Enhanced mechanism to detect and mitigate economic denial of sustainability (EDoS) attack in cloud computing environments, Int. J. Adv. Comput. Sci. Appl. 8 (2017) 51–58.
[56] T. Vissers, T.S. Somasundaram, L. Pieters, K. Govindarajan, P. Hellinckx, DDoS defense system for web services in a cloud environment, Fut. Gener. Comput. Syst. 37 (2014) 37–45.
[57] P. Shamsolmoali, M. Zareapoor, Statistical-based filtering system against DDOS attacks in cloud computing, in: 2014 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2014, pp. 1234–1239.
[58] M. Zakarya, DDoS verification and attack packet dropping algorithm in cloud computing, World Appl. Sci. J. 23 (2013) 1418–1424.
[59] A. Girma, M. Garuba, J. Li, C. Liu, Analysis of DDoS attacks and an introduction of a hybrid statistical model to detect DDoS attacks on cloud computing environment, in: in 2015 12th International Conference on Information Technology-New Generations, 2015, pp. 212–217.
[60] W. Dou, Q. Chen, J. Chen, A confidence-based filtering method for DDoS attack defense in cloud environment, Fut. Gener. Comput. Syst. 29 (2013) 1838–1850.
[61] P. Negi, A. Mishra, B. Gupta, Enhanced CBF Packet Filtering Method to Detect DDoS Attack in Cloud Computing Environment, arXiv preprint, 2013.
[62] B. Wang, Y. Zheng, W. Lou, Y.T. Hou, DDoS attack protection in the era of cloud computing and software-defined networking, Comput. Netw. 81 (2015) 308–319.
[63] H.S. Bedi, S. Shiva, Securing cloud infrastructure against co-resident DoS attacks using game theoretic defense mechanisms, in: Proceedings of the international conference on advances in computing, communications and informatics, 2012, pp. 463–469.
[64] A.K. Marnerides, P. Spachos, P. Chatzimisios, A.U. Mauthe, Malware detection in the cloud under Ensemble Empirical Mode Decomposition, in: 2015 International Conference on Computing, Networking and Communications (ICNC), 2015, pp. 82–88.
[65] R.G. Rao, M.J. Nene, SEDoS-7: a proactive mitigation approach against EDoS attacks in cloud computing, in: 2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), 2017, pp. 965–970.
[66] F. Al-Haidari, M. Sqalli, K. Salah, Evaluation of the impact of edos attacks against cloud computing services, Arab. J. Sci. Eng. 40 (2015) 773–785.
[67] D. Krishnan, M. Chatterjee, An adaptive distributed intrusion detection system for cloud computing framework, in: International Conference on Security in Computer Networks and Distributed Systems, 2012, pp. 466–473.
[68] B. Cha, J. Kim, Study of multistage anomaly detection for secured cloud computing resources in future internet, in: 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, 2011, pp. 1046–1050.
[69] C.N. Modi, D.R. Patel, A. Patel, R. Muttukrishnan, Bayesian Classifier and Snort based network intrusion detection system in cloud computing, in: 2012 Third International Conference on Computing, Communication and Networking Technologies (ICCCNT'12), 2012, pp. 1–7.
[70] S. Teng, C. Zheng, H. Zhu, D. Liu, W. Zhang, A cooperative intrusion detection model for cloud computing networks, Int. J. Secur. Appl. 8 (2014) 107–118.
[71] M. Ficco, Security event correlation approach for cloud computing, IJHPCN 7 (2013) 173–185.
[72] W.G. Morein, A. Stavrou, D.L. Cook, A.D. Keromytis, V. Misra, and D. Rubenstein, "Using graphic turing tests to counter automated DDoS attacks against web servers," in Proceedings of the 10th ACM conference on Computer and communications security, 2003, pp. 8–19.
[73] V. Varadharajan, U. Tupakula, Security as a service model for cloud environment, IEEE Trans. Netw. Serv. Manag. 11 (2014) 60–75.
[74] T. Peng, C. Leckie, K. Ramamohanarao, Protection from distributed denial of service attacks using history-based IP filtering, in: IEEE International Conference on Communications, 2003. ICC'03., 2003, pp. 482–486.
[75] J. Udhayan, R. Anitha, Demystifying and rate limiting ICMP hosted DoS/DDoS flooding attacks with attack productivity analysis, 2009 IEEE Int. Adv. Comput. Conf. (2009) 558–564.
[76] X. Chen, J. Ma, S. Li, K. Chen, A. Serhrouchni, A dynamic probabilistic marking approach with multi-tag for tracing ICMP-based DoS attacks, Wuhan Univ. J. Natl. Sci. 18 (2013) 484–488.
[77] M. Kumar, R. Udayakumar, Identifying and blocking high and low rate DDOS ICMP flooding, Indian J. Sci. Technol. 8 (2015).
[78] Y.-K. Kwok, R. Tripathi, Y. Chen, K. Hwang, HAWK: halting anomalies with weighted choking to rescue well-behaved TCP sessions from shrew DDoS attacks, in: International Conference on Networking and Mobile Computing, 2005, pp. 423–432.
[79] H. Harshita, Detection and prevention of ICMP flood DDOS attack, Int. J. New Technol. Res. 3 (2017).
[80] J. Udhayan, R. Anitha, Demystifying and rate limiting ICMP hosted DoS/DDoS flooding attacks with attack productivity analysis, in: presented at the IEEE International Advance Computing Conference, 2009.

Sayed Qaiser Ali Shah is the Ph.D. scholar at university of Engineering and Technology, Taxila, Pakistan. He received his M. S. Degree in Information Technology from National University of Science and Technology, Islamabad, Pakistan in 2014 and received Gold Medal in M.*Sc.* (Computer Science) from Institute of Management Sciences, Peshawar, Pakistan in 2011. He served as a lecturer in well reputed institutes. His-area of research is QoS and Security of Computer Networks, Software Defined Networks and Cloud Computing.

Farrukh Zeeshan Khan received his Ph.D. in Telecommunications in 2012 from Institute of Telecommunications, Vienna University of Technology, Vienna, Austria, and Masters in Computer Science in 2002 from Islamia University Bahawalpur, Pakistan. During Ph.D. studies he worked on Optical Burst Switched Network. Currently he is working as Assistant Professor in Computer Science Department, University of Engineering and Technology, Taxila, Pakistan. His-research interests include Internet of Things, Mobile Adhoc Networks, and Next generation all-optical networks.

Muneer Ahmad completed his PhD in Computer Science from Universiti Teknologi PETRONAS, Malaysia in 2014. He has 18 years of teaching, research and administrative experience internationally. Dr. Muneer Ahmad has authored numerous research papers in refereed research journals, international conferences and books. Further, he successfully completed several funded research projects. His-areas of interest include Data science, big data analysis, machine learning, bioinformatics and medical informatics.