Arab J Sci Eng (2016) 41:5037–5047
DOI 10.1007/s13369-016-2210-7

RESEARCH ARTICLE - COMPUTER ENGINEERING AND COMPUTER SCIENCE

# An Experimental Evaluation of the EDoS-Shield Mitigation Technique for Securing the Cloud

**Saeed Alsowail**[1] · **Mohammed H. Sqalli**[1] · **Marwan Abu-Amara**[1] · **Zubair Baig**[2] · **Khaled Salah**[3]

Received: 30 December 2015 / Accepted: 11 May 2016 / Published online: 26 May 2016

**Abstract** Security of cloud services is of utmost importance for contemporary cloud providers. In addition to the traditional malicious attacks that have targeted cloud datacenters in the past, new and persistent threats have changed the landscape of cyber-attacks in recent times. Economic Denial of Sustainability (EDoS) attacks are one of such variant attack types with serious implications and consequences. Such attacks exploit the scalability and elasticity characteristics of the cloud to enforce unwanted resource allocation with the aim of causing economic losses to the cloud service owner. In this paper, we present an experimental study to evaluate the effectiveness of the popular EDoS-Shield technique which is designed to mitigate EDoS attacks. The effectiveness of EDoS-Shield is studied in terms of the needed VM compute resources, response time, and CPU utilization.

**Keywords** Cloud computing · EDoS attack · EDoS-Shield · Experimental evaluation

✉ Khaled Salah
khaled.salah@kustar.ac.ae

Saeed Alsowail
alsowail@kfupm.edu.sa

Mohammed H. Sqalli
sqalli@kfupm.edu.sa

Marwan Abu-Amara
marwan@kfupm.edu.sa

Zubair Baig
z.baig@ecu.edu.au

[1] Computer Engineering Department, King Fahd University of Petroleum and Minerals, Dhahran, KSA

[2] School of Science and Security Research Institute, Edith Cowan University, Joondalup, Australia

[3] Electrical and Computer Engineering Department, Khalifa University, Abu Dhabi, UAE

## 1 Introduction

Cloud computing as a paradigm has fostered rapid convergence of computing resources and centralized and cost-effective computing resource allotment for organizations [1]. The global adoption of this paradigm can be attributed to the overall cost-effectiveness for both end users and corporations. Specifically, the National Institute of Standards and Technology definition of the cloud computing enlists five essential characteristics [2]: on-demand self-service, broad network access, resource pooling, rapid elasticity or expansion, and measured service. Many corporations shifted computing jobs as well as data to the cloud foreseeing the above-listed advantages. In view of the outsourcing of resource allocation and data storage to the cloud, security is a critical concern that requires careful planning by the cloud service providers to ascertain a robust and reliable provision of service [3]. In addition to known malicious attacks such as Denial of Service and Privilege Escalation that target traditional datacenters, new and sophisticated attacks have emerged which are custom build for the cloud.

One of the well-known adversarial attacks against critical computing resources has been the Distributed Denial of Service (DDoS) attack wherein the attacker compromises several machines on the Internet and subsequently triggers a massive burst of useful network traffic targeting a single (at times multiple) victim machines, crippling their capacity from providing further service to legitimate clients [4]. In a conventional datacenter, the DDoS attack would incapacitate the core service-providing machines from serving subscribers thus affecting production and consequently the overall business. In 2008, Hoff pointed out the economic effect of a DDoS attack that affects the cloud pricing model through exploitation of its elasticity property [5]. When a DDoS attack targets an adopter of the cloud computing technology, computing resources will be allocated dynamically to that adopter without its prior consent. The users, or customers, of the cloud computing adopter will be able to continue to use the services provided by the adopter. However, the cloud adopter automatically pays for all resources and thus allocated as a consequence of the requested resources from the attacker. Such an attack, i.e., Economic Denial of Service (EDoS), can therefore be defined as a crafted variant of a DDoS attack that targets the elasticity of the cloud to cause economic loss [5–7].

While appreciating the extent of loss a cloud provider may face through an EDoS attack, a robust and effective mechanism for mitigating its effects is critical. One of the few techniques found in the literature for mitigating an EDoS attack is the EDoS-Shield [6]. As part of our contribution presented in this paper, we propose a test methodology and perform an experimental evaluation of the EDoS-Shield under dynamic system and network conditions. Experimental analysis of EDoS-Shield helps ascertain the effectiveness of the scheme in defending against EDoS attacks against deployed cloud resources. The testbed was prepared to accurately represent the EDoS-Shield technique. The addition of network-based elements to represent individual components of EDoS-Shield facilitated acquisition and analysis of results with a high degree of accuracy. The results obtained from the experimental testbed are compared with those obtained from the simulation results provided in [6]. Results obtained from the experiments were at par with expected outcomes. An extensive analysis is provided on cloud resource utilization, EDoS-Shield performance overhead, and the false alarms generated by the scheme.

The rest of this paper is organized as follows: Sect. 2 reviews related work on EDoS attacks. In Sect. 3, the EDoS-Shield is elaborated upon. We provide a detailed description of the experimental testbed setup in Sect. 4. The experimental methodology is given in Sect. 5. Section 6 presents and discusses the results obtained from the experiments, and compares these to the simulation results of [6]. The EDoS-Shield is evaluated in an environment comprising a real-world end user of the cloud services, in Sect. 7. Finally, Sect. 8 concludes the paper and provides future directions for research.

## 2 Related Work

Detection of adversarial EDoS attacks against cloud resources and service providers remains a concern for the providers of cloud-based services. Very little research has been reported to address this problem. The mitigation techniques that have been developed to appease the effects of EDoS attacks are limited in functionality and nontrivial to realize and deploy. We highlight some of the key proposals for containing EDoS attacks as reported in the literature.

Self-verifying Proof-of-Work (sPoW) was presented as a mitigation technique to block EDoS attacks [8]. sPoW requires a proof of work (crypto-puzzles) from the clients before allowing them access to elastic cloud services hosted on the cloud servers. However, sPoW has a number of disadvantages as were discussed in [6]. The key drawback is the lack of assurance of resource access to legitimate clients, as the crypto-puzzles would vary in complexity and may pose unpredictable system behavior.

Saini and Somani [9] focused on mitigating Web site index page-based EDoS attacks. In their work, they found that the EDoS attacks have a severe impact when applied to the index pages of Web sites hosted on the cloud. The authors designed a mitigation utility called the IPA-Defender which implements various prevention models proposed by them. These prevention models were proposed based on an analysis study on the human Web browsing behavior and through state model representations of activity representative of typical Web sessions. The dynamics of such a model entail limited guarantees on the level of effectiveness of the defender module in preventing EDoS attacks.

VivinSandar and Shenai [7] showed how a DDoS attack is transformed to an EDoS attack against the cloud. They also surveyed the literature for mitigation techniques against EDoS and DDoS attacks in the cloud. Finally, they proposed a security framework for EDoS attack protection. However, Salah and Kahtani [10] pointed out that this mitigation technique is inefficient because it is based on merely the deployment of a traditional firewall for identifying network traffic anomalies to confirm an attack, without a depth analysis of the traffic patterns.

Salah and El-Badawi [13] proposed a mitigation technique for the EDoS attack using an in-cloud scrubber service. Their solution is provided as one of the services by the cloud service provider. The solution has two modes of operation, namely *normal* and *suspected*. When the workload of the Web server is per expectation, the system will operate in the normal mode. But when the service provider notices that the traffic directed toward the Web server exceeds an acceptable threshold, the operation will be switched to the suspected mode. In this mode, the requests will be sent to a scrubber server which will subsequently challenge clients with puzzles. The scheme proposed is similar to the one in [6], albeit with added capabilities to detect low-intensity DDoS attacks.

Yu et al. [12] proposed a dynamic resource allocation strategy to counter DDoS attacks in a cloud computing environment. In their work, they dynamically allocate unused resources, as needed, to create several intrusion prevention system (IPS) instances that block a DDoS attack targeting a specific cloud adopter, while ensuring quality of service (QoS) for end users. The authors focused on how to dynamically allocate resources for the mitigation technique. They validated their work through mathematical modeling.

Koduru et al. [13] proposed a mitigation technique for the EDoS attack based on the time spent on a Web page (TSP) for detecting the attack. The authors discussed that the TSP for the attack traffic differs from the mean TSP of the traffic of legitimate users accessing the page. They analyzed the deviation of the TSP for malicious traffic from the mean TSP of the legitimate traffic to identify EDoS attacks.

Alosaimi and Al-Begain [14] proposed a mitigation system for the EDoS attack based on the EDoS-Shield. Their mitigation system, namely the DDoS-Mitigation System (DDoS-MS), improves upon the EDoS-Shield mitigation technique by decreasing the end-to-end latency perceived by the end users of the cloud service. In addition to the firewall and the verifier node components of the EDoS-Shield, the DDoS-MS also has a client puzzle server, a DNS server, and several green nodes. The scheme proposed was not tested by the authors for performance.

Sqalli et al. [6] proposed a solution, namely EDoS-Shield to mitigate the effects of an EDoS attack. The EDoS-Shield classifies client requests into whitelists and blacklists based on the source of the requests, thus confirming whether the origin of the requests is either legitimate or bot based. This is achieved using a verifier node which creates and maintains the white and blacklists. A virtual firewall is implemented to block all the requests that originate from the blacklisted sources. This work was expanded by Al-Haidari et al. [15] to mitigate the attack in case the attacker uses spoofed IP addresses. The next section provides a detailed elaboration of the EDoS-Shield scheme.

Al-Haidar et al. [16] proposed an analytical model for studying the impact of EDoS attacks against single-class cloud services comprising a single type of application service at the datacenter. The model itself included three key performance metrics, namely user response time (end-to-end), computing resource utilization, and the total cost incurred through an attack. Simulations were carried out to compare against results obtained from the analytical model derived.

In [17], the authors presented the argument that multi-tenant physical servers hosting a large number of virtual machines would be reliant on several components of the cloud architecture. Through a proposed systemic model of the infrastructure cloud, the effects of an EDoS attack can be best quantified. Specifically, the size of the cloud, application instances, resource allocation strategies, and the strength and duration of EDoS attacks are to be considered while setting up the experimental testbed. A detailed analysis on the effect of variation of the above parameters was studied and reported.

Masood et al. [18] presented the EDoS Armor mitigation framework against EDoS attacks. The presented framework mitigates the EDoS attack effects against e-commerce applications running on the cloud. Once a client has successfully authenticated against the cloud services, the framework controls the rate of client request arrival at the server through the maintenance of a client priority table that records the priority levels of individual clients. The priority levels vary based on the request types made by the client, with lower priority clients made to wait longer for service access. While the scheme does protect the server resources against the effects of the EDoS attack, no guarantees on the service times can be provided to the clients.

## 3 The EDoS-Shield Mitigation Technique

Figure 1 shows the architecture of the EDoS-Shield mitigation technique as presented in [6]. The main components of the EDoS-Shield mitigation technique are the virtual firewall (VF) and the verifier node (V-Node). The virtual firewall has two lists of IP addresses, whitelist and blacklist. The whitelist consists of those source IP addresses which are considered legitimate. All the requests that originate from these sources are allowed to pass through the firewall to the cloud adopter servers. On the other hand, all the IP addresses that are contained in the blacklist are considered malicious, and hence, all the traffic that arrives from these IPs is automatically blocked by the firewall.

When there is a request from an unknown source, i.e., its IP address is not found in the firewall's lists, the request is forwarded to the V-Node which subsequently sends a graphical Turing test to the source of this request. If the request has been issued by a human user, a correct response to the Turing test is highly likely. In such a case, the V-Node will add the IP address of the source of the request to the whitelist of the firewall. Any following requests from this source will subsequently be considered as legitimate and will be allowed to pass through the firewall. However, if the request has been generated by a machine, e.g., bot, the machine will most likely fail to solve the Turing test. In this case, the V-Node will

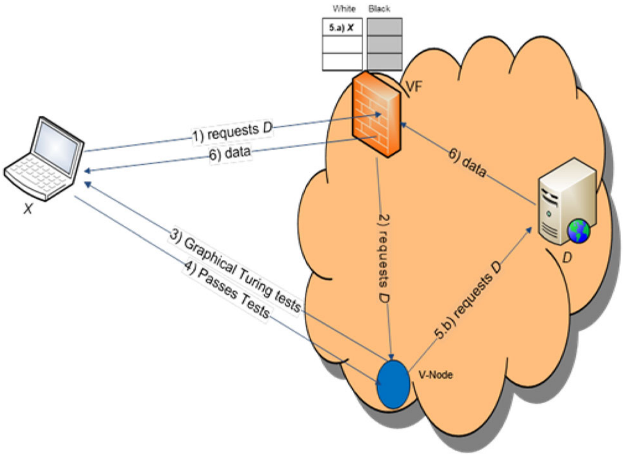

**Fig. 1** EDoS-Shield architecture

add the IP address of the source of the request to the blacklist of the firewall. Any following requests from this source will be blocked by the firewall. As an essential outcome of the graphical Turing test, a clear differentiation between human and machine users is made by the scheme. The level of difficulty of the graphical Turing test is kept constant, and the white and blacklists are not reset anytime during operations of the cloud provider. However, through intervention of the provider's administrators, the scheme can be reinitiated so that legitimate clients who frequently fail the Turing test are also considered for inclusion into the whitelist.

The improvements made to the EDoS-Shield as reported in [14] are elaborated as follows. The first two requests that originate from a given client are tested to establish the legitimacy of the source. The first request is treated the same way as with the EDoS-Shield. The source IP address is added to the white or blacklist of the firewall based on the outcome of the graphical Turing test. The second request will be forwarded to the client puzzle server if its source IP address is listed in the whitelist of the firewall, and its time-to-live (TTL) value matches the acceptable TTL value as stored in the firewall. Otherwise, the packet will be dropped. For the second request, a crypto-puzzle will be sent to the client. If the client passes the test, the puzzle server will send a positive acknowledgement to the firewall. Thereafter, all subsequent requests originating from this source will be forwarded to the DNS server and then to the cloud servers via front-end servers, which are deployed to shield the cloud servers. This process verifies the legitimacy of the source. If the source IP address of the second request is listed in the blacklist of the firewall, and the TTL value or the timestamp of this request (or included packets) matches TTL values of packets of the first request, the firewall will drop this request as well as all subsequent requests originating from this source. If the TTL values do not match, the firewall will forward the request to the verifier node and the legitimacy of the source will be tested again. The authors improved their mitigation system in [19].

Considering the infancy of the schemes proposed in [14] and [19], without any backing through experimental results, we have proceeded with conducting experiments on the EDoS-Shield mitigation technique of [6]. The next section discusses the experimental testbed setup and the steps followed to evaluate the effectiveness of the EDoS-Shield in mitigating EDoS attacks. We subsequently elaborate upon the results collected from the testbed and do a formal discussion and comparison with those reported in [6].

## 4 Testbed Setup

This section discusses the EDoS-Shield mitigation technique deployment and subsequent evaluation in a laboratory environment. Since the main objective of this work is to compare the results obtained from the experimental testbed to those obtained from simulation in [6], we prepared the testbed to be very close to the assumptions made in the simulation. As a first experiment, the testbed was deployed without the mitigation technique in place, in order to study the effect of the EDoS attack on the cloud resources. Figure 2 shows the architecture of the testbed before implementing the EDoS-Shield mitigation technique. The second experiment was an actual implementation of the EDoS-Shield mitigation technique, so as to study its effect in blocking the EDoS attack. Figure 3 shows the architecture of the testbed including EDoS-Shield. The results obtained from the testbed are compared to those obtained from the simulation in the next section for both cases. The main components of the testbed for each case will be discussed in following subsections.

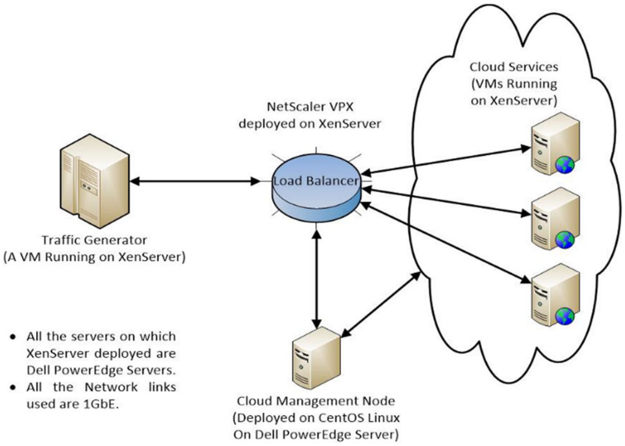
Cloud Services
(VMs Running
on XenServer)
NetScaler VPX
deployed on XenServer
Load Balancer
Traffic Generator
(A VM Running on XenServer)
All the servers on which XenServer deployed are Dell PowerEdge Servers.
All the Network links used are 1GbE.
Cloud Management Node
(Deployed on CentOS Linux
On Dell PowerEdge Server)

**Fig. 2** Testbed without the EDoS-Shield mitigation technique

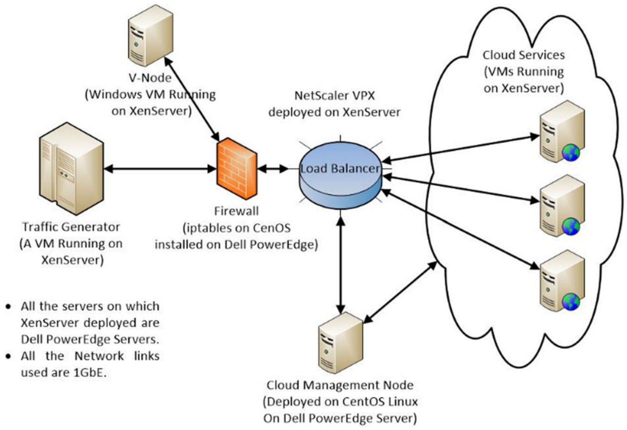

**Fig. 3** Testbed with the mitigation technique implemented

### 4.1 Cloud Services

The main component of our testbed is the cloud. Citrix's CloudPlatform [20] and XenServer [21] were used to deploy the cloud. The CloudPlatform is a cloud management software which is responsible for managing the cloud and its resources. A single physical server was deployed as a management node on which the Citrix CloudPlatform was installed. Three physical servers were designated as compute nodes on which the hypervisor, i.e., XenServer, was setup. The virtual machines (VMs), or the instances on which the services provided by the cloud are deployed, were made to run on these compute nodes. All the VMs are identical small instances that were created from a single template. This template comprises a simple Web server configured with the CentOS Linux operating system [22]. Apache Server was used as the Web server to receive client requests, upon successfully passing the Turing test for those clients which were originally blacklisted [23].

### 4.2 Load Balancer

The load balancer provides for an even distribution of network traffic, i.e., client requests among the active VMs of the cloud. We implemented the Citrix NetScaler VPX (200) [24] load balancer, which is a virtual appliance deployed as a virtual machine on XenServer, on a separate physical server. NetScaler is configured and managed through the CloudPlatform. It is the entry point to the cloud services, and hence, all the traffic that ingresses or egresses the cloud thus passes through it. The NetScaler dashboard was utilized for continuous monitoring of experimental activity.

### 4.3 Traffic Generator

The Apache JMeter HTTP traffic generator [25] was deployed for generating both legitimate and malicious network traffic. In addition to the basic features that come with JMeter by default, we added the standard set of plugins [26], in order to customize the client requests generated for delivery to the cloud service provider. JMeter was deployed on 8 VMs running on the Citrix XenServer, on a separate physical server.

### 4.4 Firewall

The firewall is an integral component of the EDoS-Shield, required for interception and comparison of all arriving requests for cloud resource access. The traffic that arrives from the whitelisted sources is allowed to access the cloud services, whereas the traffic that arrives from blacklisted sources is dropped. We implemented the Linux IPTables firewall in our testbed [27]. The firewall was configured to forward all traffic arriving from *suspected* IP addresses to the V-Node. All legitimate, i.e., whitelisted requests are forwarded to the load balancer. Both black and whitelists are regularly updated by the verifier node.

### 4.5 Verifier Node

The verifier node (V-Node) is deployed as a Web server that sends CAPTCHA requests to the clients to differentiate between humans and machines making cloud access requests. It is also responsible for updating the white and blacklists on the firewall based on responses received from the CAPTCHA tests. We implemented the V-Node using the WampServer [28] installed on a Windows VM running on a separate physical server. The CAPTCHA was implemented using the code published in [29].

## 5 Experiments Methodology

This section describes the methodology applied for the experimental runs. In the following subsection, we provide the details of performing the experiments to study the effect of the EDoS attack on the cloud before using the EDoS-Shield mitigation technique. Following which, we discuss the experimental steps executed to evaluate the effectiveness of the EDoS-Shield in blocking EDoS attacks.

### 5.1 Studying the Effect of the EDoS Attack on Cloud Computing

In order to study the effect of the EDoS attack on cloud computing, we performed a set of experiments without using the EDoS-Shield. The results of these experiments helped quantify the effect of the EDoS attack on the cloud CPU utilization and the end user-perceived response times. These results are also compared to the simulation results obtained in [6]. Figure 2 illustrates the testbed architecture used in these experiments. The traffic generator component was configured to send the traffic directly to the load balancer. The load balancer further forwards the traffic to the VM instances of the cloud on which a simple Web page is hosted. The VM instances were configured to handle 100 HTTP requests per second, with each packet of a response message being 580 bytes in length. The trigger condition to autoscale VM resources was kept the same as reported in [6], with 80 % CPU utilization marked as the upperbound. The implication here is that a new VM instance will be created and assigned to the load balancer if the cumulative CPU utilization for all the VM instances exceeds 80 % at any given time.

We placed an upperbound of 56 on the maximum number of VM instances that can be operational at any point in time. The maximum rate of attack traffic was configured to be 4000 Req/S. Before initiating an experimental run, all the cloud instances are configured to be reachable by the load balancer. The number of the instances of VMs active at any given time in the NetScalar deployment is directly proportional to the

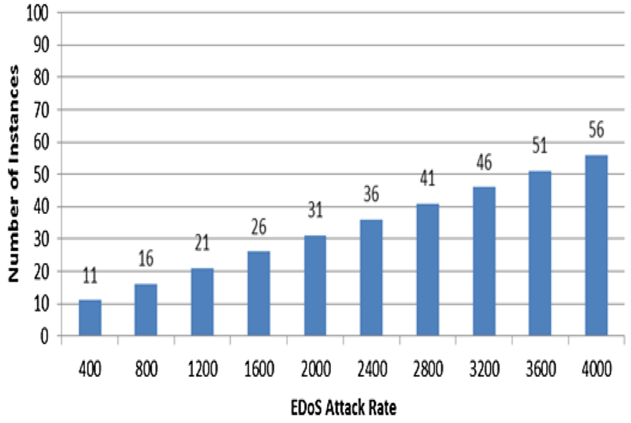

**Fig. 4** Number of required instances to keep CPU utilization below 80 % before using the EDoS-Shield

intensity of the attack at any point in the experiment. To ensure that the incoming traffic to the cloud will not utilize in excess of 80 % of the cloud resources, i.e., VM instances, we increased the number of instances following the same approach as was used in the simulation. Hence, the number of the required instances at any point time is theoretically calculated as follows:

$$\frac{\lambda}{S\mu} \leq 0.8. \text{ Thus, } S = \lceil 1.25 \times \lambda/\mu + 1 \rceil \qquad (1)$$

where $S$ is the required number of instances, $\lambda$ is the traffic arrival rate, and $\mu$ is the service rate.

In addition to the EDoS attack rate, we inject background network traffic at the rate of 400 requests per second, to emulate legitimate clients. Figure 4 shows the number of required VM instances before using the EDoS-Shield for each experiment based on Eq. 1.

After making sure that the appropriate number of instances has been assigned to the load balancer, we run the experiment by starting the traffic generation on JMeter. There are two ways to monitor traffic and to ascertain that it is received by the cloud instances. The first method to monitor the traffic is via the dashboard of NetScaler, which shows the aggregated traffic rate for all the traffic generated. The second method is via the Server Hits per Second plugin of JMeter. This method is used to check whether all the HTTP requests created from a specific traffic generating VM arrive at the cloud.

We keep monitoring the CPU utilization through Citrix's XenCenter [30], which is installed on a laptop to collect the results. When the CPU utilization of the instances reaches the steady state, the CPU utilization of each instance is collected separately, and then, the average CPU utilization is calculated. The response time is measured using Firebug add-on [31] that is installed on a Firefox Web browser. For each experiment, the response time is collected 30 times, and then, the average is calculated.

For each rate of the EDoS attack, the experiment is repeated ten times. Each time the CPU utilization and the response times are collected. After collecting the results for all the ten repetitions, the average CPU utilization and the average response times are calculated.

This subsection explained the steps followed when performing the experiments to study the effect of varying intensity EDoS attacks on cloud resources. The next subsection discusses the steps followed when performing the experiments to evaluate the EDoS-Shield mitigation technique.

### 5.2 Studying the Effectiveness of the EDoS-Shield Mitigation Technique

In this section, the experiments performed to evaluate the effectiveness of the EDoS-Shield in mitigating the EDoS attacks are discussed. Most of the steps are the same as described in the previous subsection. The novelty in these experiments is the introduction of the firewall and the V-Node, which are the components of the EDoS-Shield. Figure 3 shows the architecture of the testbed after implementing the EDoS-Shield.

In this set of experiments, the firewall is the entry point to the cloud instead of the load balancer. All the traffic that arrives at the cloud, or egresses the cloud passes through the firewall. JMeter was deployed on each of the eight traffic generator VMs which are configured to send the traffic to the firewall. We assumed the following for the traffic generator VMs when performing the experiments:

- From the eight traffic generator VMs, two will simulate the legitimate traffic, while the other six VMs will simulate the malicious traffic, and
- To emulate legitimate and rogue clients, we tag a VM as legitimate or malicious. The first request from that VM will be sent using its Web browser. The CAPTCHA will be answered correctly if it is a legitimate VM and incorrectly if it is a malicious VM.

In all the experiments, the dashboard of NetScaler shows that only the legitimate traffic arrives at the cloud. Since the legitimate traffic is configured to arrive at the rate of 400 requests per second, the number of cloud instances that are used in all the experiments is equal to 6, as indicated by equation 1. This is illustrated in Fig. 5.

The experiment for each of the EDoS rates is repeated ten times. The results are collected and calculated the same way as explained in the previous subsection. In the next section, the results of the experiments will be presented and discussed.

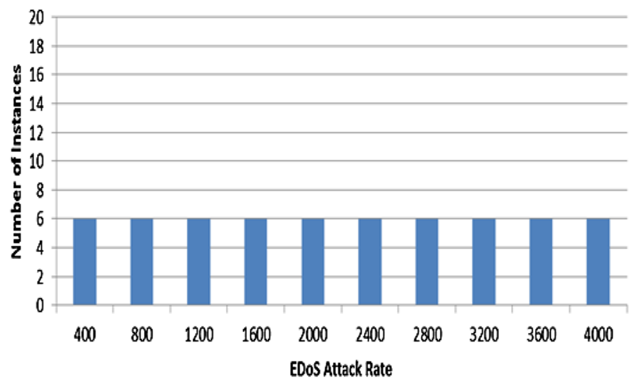

**Fig. 5** Number of instances required to keep CPU utilization below 80 % after using the EDoS-Shield

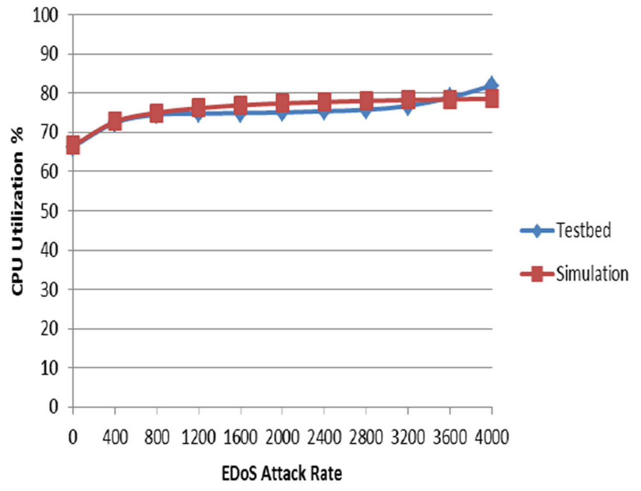

**Fig. 6** Comparison of the average CPU utilization taken for all instances before using the EDoS-Shield

## 6 Results and Discussion

### 6.1 The Effect of the EDoS Attack on Cloud Computing

The results presented in this subsection illustrate the effect of the EDoS attack on cloud resources. These results are also compared to the simulation results obtained in [6]. Figure 6 compares the average CPU utilization results obtained from the testbed to those obtained through simulation. As may be noted, the CPU utilization results of the testbed are at par with the results of the simulation. Both results show that when the intensity of an EDoS attack increases, the CPU utilization increases. Increasing number of VM instances will be added to the cloud, based on its elasticity property, as the attack rate increases. The addition of the new instances simply to handle the attack requests will result in a severe economic loss for the cloud adopter, which effectively is the purpose of the attack.

In Fig. 7, we report the relative error (false alarms) in the CPU utilization between the two findings. The highest error of 5 % was noted for the case with 4000 VM instances. Overall, the results obtained from the testbed are very close to the results obtained from the simulation in terms of CPU utilization.

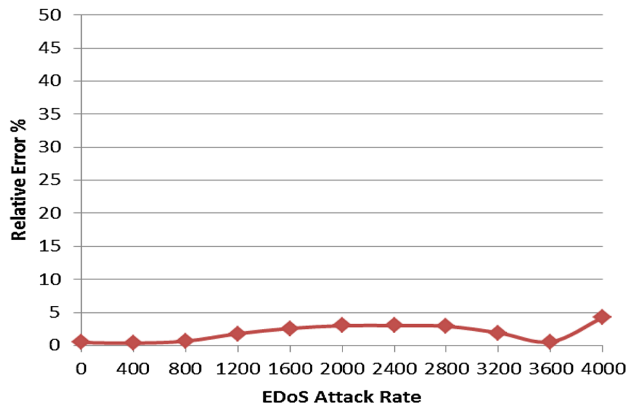

**Fig. 7** Relative error % for CPU utilization comparison before using the EDoS-Shield

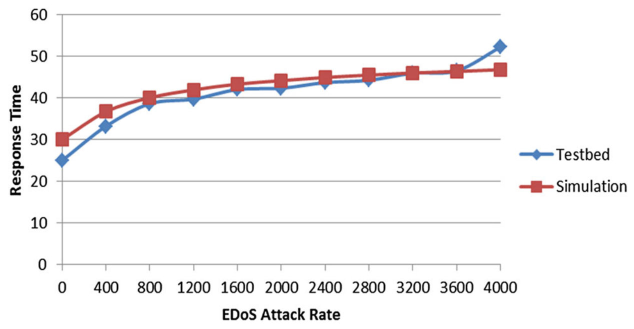

**Fig. 8** Response time (ms) comparison before using the EDoS-Shield

Figure 8 shows a comparison between the results obtained for the user-perceived response times for both the testbed and the simulation. Results obtained through experiments were found to be at par with those from simulation. It is evident from the results reported here that the EDoS attack does indeed cause unwanted delays for legitimate clients, and therefore, a scheme for countering this attack is mandatory. When the rate of the EDoS attack increases, the response time approaches an unacceptable range for end users (in excess of 45 ms for higher numbers of VM instances). In addition, if the resource allocation policy does auto-instantiation of new VM instances in response to increasing delays, the effect of the EDoS attack is higher.

Figure 9 shows the relative error percentage for the response time comparisons reported in Fig. 8. Although the figure shows that the difference between the results of some of the experiments is around 16 %, the maximum difference between the response time results of the testbed and the simulation is around 5 ms.

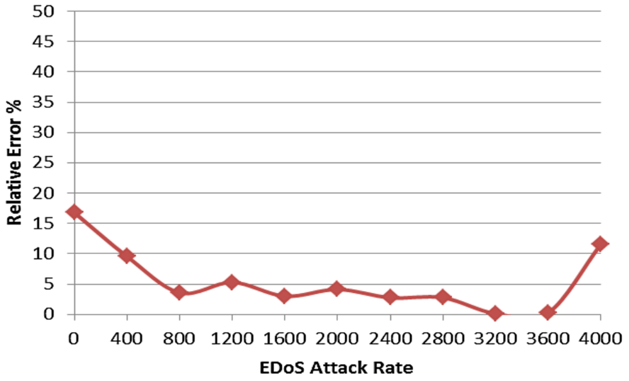

**Fig. 9** Relative error % for the response time comparison before using the EDoS-Shield

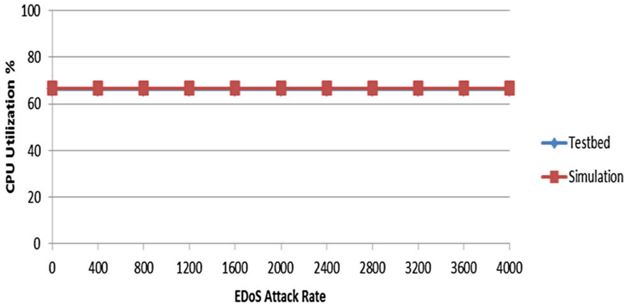

**Fig. 10** Comparison of the average CPU utilization taken for all instances after using the EDoS-Shield

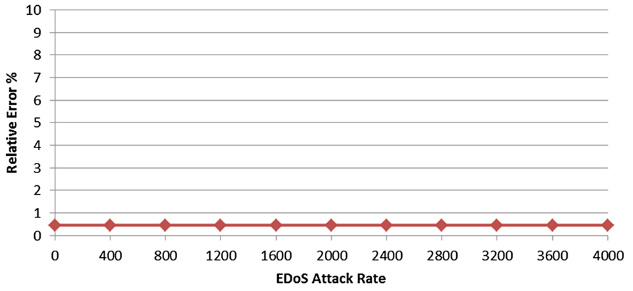

**Fig. 11** Relative error % for the CPU utilization comparison after using the EDoS-Shield

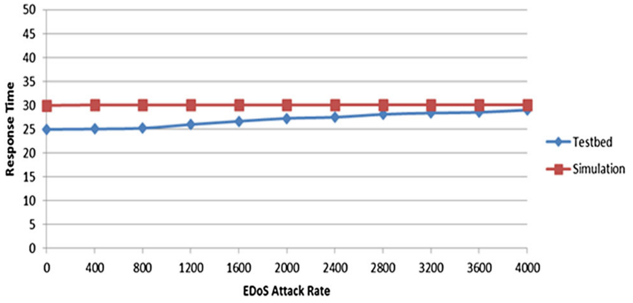

**Fig. 12** Response time (ms) comparison after using the EDoS-Shield

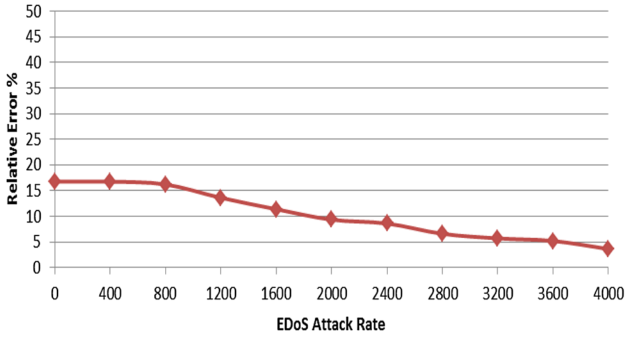

**Fig. 13** Relative error % for the response time comparison after using the EDoS-Shield

### 6.2 The Effect of Using the EDoS-Shield

After studying the effect of the EDoS attack on cloud resources, the results of adding the EDoS-Shield mitigation technique will be discussed in this subsection.

After implementing and using the EDoS-Shield mitigation technique on the testbed, it was evident from the NetScalar analysis that only the traffic generated from legitimate VM traffic generators was allowed to arrive at the cloud instances. All suspected traffic was thus prevented from reaching the cloud VMs, through packet dropping at the firewall. For this reason, the CPU utilization for both the testbed and simulation is near constant during all the experiments, as shown in Fig. 10. Both the results of the testbed and the simulation illustrate that the EDoS-Shield is capable of eliminating the effect of the EDoS attacks, as represented by CPU utilization.

The relative error percentage for the CPU utilization comparison when using the EDoS-Shield is calculated and presented in Fig. 11. As is evident from the figure, the relative error percentage is always below 1 %.

Figure 12 shows a comparison between the values of response times for both the testbed and the simulation when the EDoS-Shield mitigation technique is applied. The difference between the results of the testbed and the simulation is minimal, as presented in Fig. 12. The slight increase in the values of response times as the EDoS attack rate increases, which is clear in the results of the testbed, is through the added processing of packets that takes place at the firewall.

Figure 13 shows the relative error percentage for the response time comparison between the testbed and the simulation when the EDoS-Shield mitigation technique is used. The maximum difference in the relative error percentage occurs in the first experiment. The percentage is around 16 % which translates to a difference of about 5 ms.

## 7 Experiments for a Real-Life Scenario

The testbed setup discussed in Sect. 5 was prepared to validate the simulation results presented in [6]. In this section, we study the effect of the EDoS attack in an environment

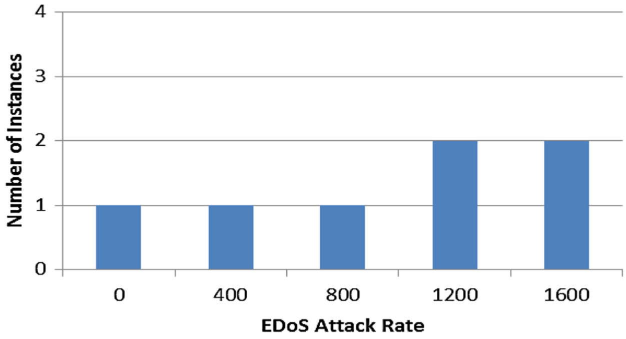

**Fig. 14** Number of required instances to keep CPU utilization below 80 % before using the EDoS-Shield

that is very close to a real-life scenario. We also study the effect of using the EDoS-Shield in such environment. The testbed setup used for these experiments is the same as that used to validate the simulation results. The only difference is the replacement of the template. The new template comprises a real Web site that has an index page with text and several graphical images [32]. The index page contains 24 elements that are downloaded to the browser of the client upon access, total size being 507.4 KB. We added an additional picture to the index page to achieve this size and to closely represent a standard Web site.

Based on experimental analysis, it was noted that setting the upper bound on CPU utilization for triggering autoscaling to 40 % was the most optimal. This analysis was based on the fact that nearly 40 % CPU utilization is achieved when nearly 1600 HTTP requests per second are generated. However, we found that NetScaler will not allow more than 2100 HTTP requests per second to pass through it in these experiments since the throughput at these rates will exceed the limit permitted by the current license of NetScaler. This limitation was perceived as a consequence of the large size of files hosted on the Web site, i.e., 507.4 KB as opposed to 580 Byte files used in the previous experiments. Therefore, we performed experiments for EDoS attack rates (per second) of 0, 400, 800, 1200, and 1600, respectively. In addition, background traffic of 400 requests per second is generated to represent legitimate traffic. Hence, the maximum traffic rate is equal to 2000 requests per second.

The maximum number of instances used in the experiments before using the EDoS-Shield was 2, while the number of instances is always equal to 1 with EDoS-Shield. This is shown in Figs. 14 and 15, respectively.

The following two subsections present and discuss the results of the experiments.

### 7.1 The Effect of the EDoS Attack on Cloud Computing

Figure 16 shows the CPU utilization before using the EDoS-Shield. It is clear that the CPU utilization increases in the first three experiments as the attack rate increases. Subsequently, it drops to 20 % for an attack rate of 1200 requests per second. At this attack rate, the total traffic rate is 1600 bytes, including legitimate traffic. This will result in around 40 % CPU utilization. As a result, a new instance is created, and the CPU utilization will decrease by nearly a half, as shown in Fig. 16. The CPU utilization increases again with increasing attack rate to 1600 requests per second. This behavior is similar to the results reported in Sect. 6 thus illustrating that increasing computing resources will be allocated to the cloud as the EDoS attack rate increases. This is because the CPU utilization increases as the EDoS attack rate increases.

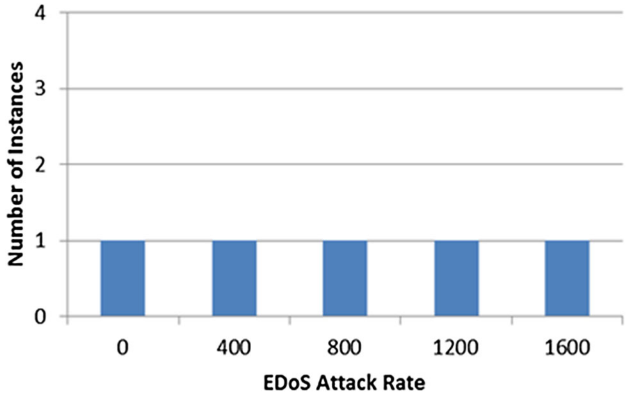

**Fig. 15** Number of required instances to keep CPU utilization below 80 % after using the EDoS-Shield

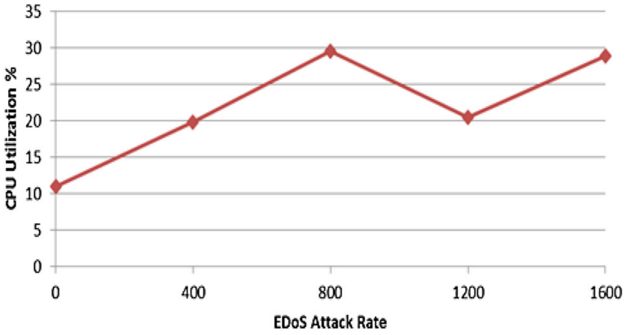

**Fig. 16** CPU utilization before using the EDoS-Shield

Figure 17 shows the response times before using the EDoS-Shield. As shown in the figure, the EDoS attack has a severe effect on the response time if no mitigation technique is in place. This behavior is the same as discussed previously in Sect. 6.

### 7.2 The Effect of Using the EDoS-Shield

Figure 18 presents the results of the CPU utilization after using the EDoS-Shield. The CPU utilization is almost fixed, and the number of instances is always 1. The dashboard of NetScaler shows that only the legitimate traffic can access

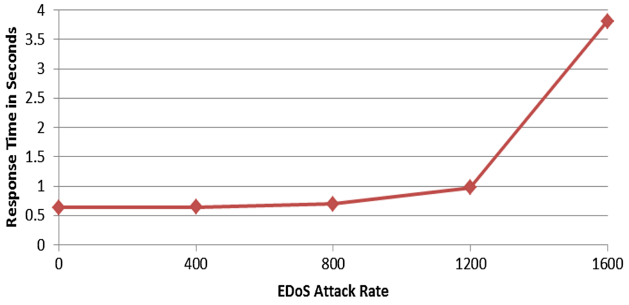

**Fig. 17** Response time before using the EDoS-Shield

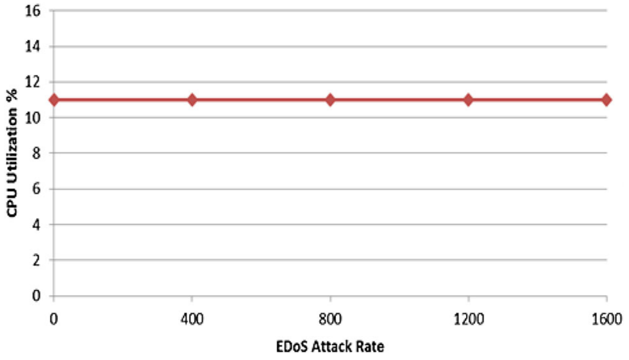

**Fig. 18** CPU utilization after using the EDoS-Shield

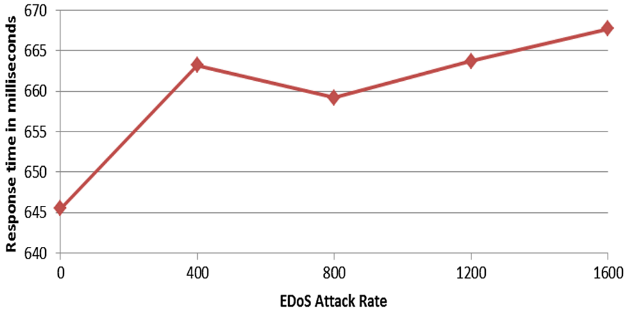

**Fig. 19** Response time after using the EDoS-Shield

the cloud services. This behavior is the same as that reported in Fig. 10.

In Fig. 19, the results of the response time after using the EDoS-Shield are presented. The response time increases slightly as the attack rate increases because of the packet processing delays imposed by the firewall. However, this increase is significantly below the results of the response time of Fig. 17 when no mitigation technique is used at all. This is clearly illustrated in Fig. 20, which shows the comparison between the response times before and after using the EDoS-Shield.

The results of Sect. 7 confirm the results of Sect. 6 in illustrating the significant effect of the EDoS attack on both the CPU utilization and the response time. The results presented in both Sects. 6 and 7 also confirm that the use of the EDoS-Shield can significantly minimize the effect of the EDoS attack on cloud resources.

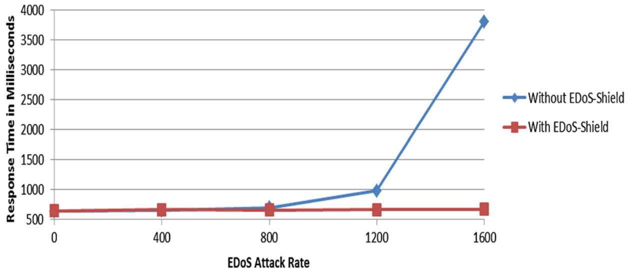

**Fig. 20** Response time comparison before and after using the EDoS-Shield

## 8 Conclusion and Future Work

In this paper, we have conducted an experimental study to evaluate and measure the effectiveness of the EDoS-Shield technique to mitigate and counter EDoS attacks. Our experimental results show that the EDoS attack does indeed exploit the elasticity property of the cloud and consequently leads to illegitimate resource consumption and economic loss for the cloud service owner. Experimental results showed that the impact of and EDOS attack can be detrimental to the performance of a cloud service in terms of the exhaustion of CPU power of VM compute resources as well as a large increases in response time. The EDoS-Shield was implemented on a testbed to evaluate its effectiveness against EDoS attacks. Measured results have shown that the EDoS-Shield is highly effective in mitigation of attacks, in which minimal VM compute resources were allocated while at the same time acceptable performance (in terms of response time and CPU utilizations) was attained.

**Acknowledgments** The authors would like to acknowledge the support provided by King Abdulaziz City for Science and Technology (KACST) through the Science and Technology Unit at King Fahd University of Petroleum and Minerals (KFUPM) for funding this work through Project No. 11-INF1609-04 as part of the National Science, Technology and Innovation Plan.

## References

1. Fox, A.; Griffith, R.; Joseph, A.; Katz, R.; Konwinski, A.; Lee, G.; Patterson, D.; Rabkin, A.; Stoica, I.: Above the clouds: a Berkeley view of cloud computing. Dep. Electr. Eng. Comput. Sci. Univ. Calif. Berkeley Rep. UCBEECS **28**, 13 (2009)
2. Mell, P.; Grance, T.: The NIST definition of cloud computing. Natl. Inst. Stand. Technol. **53**(6), 50 (2009)
3. IDC eXchange? “Blog Archive?” New IDC IT Cloud Services Survey: Top Benefits and Challenges

4. Baig, Z.; Salah, K.: Multi-agent pattern recognition mechanism for detecting distributed denial of service attacks. IET Inf. Secur. **4**(1), 333–343 (2010)
5. Hoff, C.: The economic denial of sustainability concept. http://rationalsecurity.typepad.com/blog/2008/11/index.html
6. Sqalli, M.H.; Al-Haidari, F.; Salah, K.: EDoS-shield-a two-steps mitigation technique against edos attacks in cloud computing. In: 2011 Fourth IEEE International Conference on Utility and Cloud Computing (UCC), pp. 49–56 (2011)
7. VivinSandar, S.; Shenai, S.: Economic denial of sustainability (edos) in cloud services using http and xml based ddos attacks. Int. J. Comput. Appl. **41**(20), 11–16 (2012)
8. Khor, S.H.; Nakao, A.: Spow: on-demand cloud-based eddos mitigation mechanism. In: HotDep (Fifth Workshop on Hot Topics in System Dependability) (2009)
9. Saini, B.; Somani, G.: Index page based EDoS attacks in infrastructure cloud. In: Recent Trends in Computer Networks and Distributed Systems Security, pp. 382–395 (2014)
10. Salah, K.; Kahtani, A.: Performance evaluation comparison of snort NIDS under Linux and Windows Server. Int. J. Netw. Comput. Appl. **33**(1), 6–15 (2010)
11. Kumar, M.N.; Sujatha, P.; Kalva, V.; Nagori, R.; Katukojwala, A.K.; Kumar, M.: Mitigating economic denial of sustainability (EDoS) in cloud computing using in-cloud scrubber service. In: 2012 Fourth International Conference on Computational Intelligence and Communication Networks (CICN), pp. 535–539 (2012)
12. Yu, S.; Tian, Y.; Guo, S.; Wu, D.: Can we beat DDoS attacks in clouds? IEEE Trans. Parallel Distributed Sys. **25**(9), 2245–2254 (2014)
13. Salah, K.; El-Badawi, K.: Performance evaluation of interrupt-driven kernels in gigabit networks. In: Proceedings of the 2003 IEEE Conference on Global Telecommunications, (IEEE GLOBECOM 2003), San Francisco, USA, pp. 3953-3957, 1–5 Dec (2003)
14. Alosaimi, W.; Al-Begain, K.: A new method to mitigate the impacts of the economical denial of sustainability attacks against the cloud. In: Proceedings of the 14th Annual Post Graduates Symposium on the convergence of Telecommunication, Networking and Broadcasting (PGNet), pp. 116–121 (2013)
15. Al-Haidari, F.; Sqalli, M.H.; Salah, K.: Enhanced EDoS-Shield for mitigating EDoS attacks originating from spoofed IP addresses. In: 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), pp. 1167–1174 (2012)
16. Al-Haidari, F.; Sqalli, M.H.; Salah, K.: Evaluation of the Impact of EDoS Attacks Against Cloud Computing Services. Arab. J. Sci. Eng. **40**, 773–785 (2015)
17. Somani, G.; Gaur, M.; Sanghi, D.: DDoS/EDoS attack in cloud: affecting everyone out there!. In: SIN (2015)
18. Masood, M.; Anwar, Z.; Raza, S.; Hur, M.: EDoS armor: a cost effective economic denial of sustainability attack mitigation framework for e-commerce applications in cloud environments. In: 2013 16th International Multi Topic Conference (INMIC) (2013)
19. Alosaimi, W.; Al-Begain, K.: An enhanced economical denial of sustainability mitigation system for the cloud. In: 2013 Seventh International Conference on Next Generation Mobile Apps, Services and Technologies (NGMAST), pp. 19–25 (2013)
20. CloudPlatform—Cloud Orchestration to support Infrastructure-as-a-Service. http://www.citrix.com/content/citrix/en_us/products/cloudplatform/overview.html
21. Citrix XenServer—Efficient Server Virtualization Software. http://www.citrix.com/content/citrix/en_us/products/xenserver/overview.html
22. CentOS—The Community ENTerprise Operating System. http://www.centos.org/
23. Apache HTTP Server. http://httpd.apache.org/
24. NetScaler Application Delivery Controller–Application security and cloud scalability. http://www.citrix.com/content/citrix/en_us/products/netscaler-application-delivery-controller/overview.html
25. Apache JMeter. http://jmeter.apache.org/
26. Custom Plugins for Apache JMeter. JMeter-Plugins.org. http://jmeter-plugins.org/
27. netfilter—firwalling, NAT, and packet mangling for linux. http://www.netfilter.org/
28. WampServer. http://www.wampserver.com/en/
29. Secureimage—PHP CAPTCHA. http://www.phpcaptcha.org/
30. XenCenter. http://www.xenserver.org/overview-xenserver-open-source-virtualization/download.html
31. Firebug. http://getfirebug.com/
32. 000Webhost.com. http://www.000webhost.com/templates/Games/template_45