Arab J Sci Eng (2015) 40:773–785
DOI 10.1007/s13369-014-1548-y

RESEARCH ARTICLE - COMPUTER ENGINEERING AND COMPUTER SCIENCE

Evaluation of the Impact of EDoS Attacks Against Cloud Computing Services

F. Al-Haidari · M. Sqalli · K. Salah

Received: 29 May 2014 / Accepted: 30 November 2014 / Published online: 27 December 2014

Abstract Cloud computing is currently one of the fastest growing segments of IT. To date, and according to a recent survey conducted by the International Data Corporation, security is the biggest challenge to cloud computing. A cloud introduces resource-rich computing platforms, where adopters are charged based on the usage of the cloud's resources, known as "pay-as-you-use" or utility computing. However, a conventional Distributed Denial-of-Service (DDoS) attack on server and network resources compromises cloud computing services by charging cloud adopters more cost due to the attack activities that consume cloud's resources. In such case, the main goal of such attack is to make the cloud computing unsustainable by targeting the cloud adopter's economic resources. Thus, it constitutes a new breed of DDoS attacks, namely Economic Denial of Sustainability (EDoS) attack. In this paper, we study the impact of EDoS attacks on the cloud computing services, considering only a single class of service. We developed an analytical model verified by a simulation model to study such impact of EDoS attacks on the cloud computing. The analytical model relies on the queuing model that captures the cloud services and considers a number of performance and cost metrics including end-to-end response time, utilization of computing resources, throughput, and the incurred cost resulting from the attack.

Keywords Cloud computing · EDoS attacks · Utility computing · Modeling and analysis

F. Al-Haidari (✉)
Computer Information System Department,
University of Dammam (UoD), Dammam, Saudi Arabia
e-mail: faalhaidari@ud.edu.sa

M. Sqalli
Computer Engineering Department, KFUPM, Dhahran,
Saudi Arabia
e-mail: sqalli@kfupm.edu.sa

K. Salah
Electrical and Computer Engineering Department,
Khalifa University of Science, Technology and Research (KUSTAR),
Sharjah, UAE
e-mail: khaled.salah@kustar.ac.ae

1 Introduction

Cloud computing continues to evolve as one of the most hyped information technology areas and has become the fastest growing segment of IT industry. Due to the flexibility, pay per use, elasticity, scalability, and other attributes promised by this paradigm, it has gained the interest of large organizations and corporates for hosting their services onto the cloud. Gartner has identified cloud computing as the first of the top 10 technologies with the potential for a significant impact on organizations for few years to come [1].

Cloud computing is designed to scale computation resources and servers in magnitude and availability based on the demand and the requested usage by end users. Moreover, adopters of the cloud service model are charged based on a pay-per-use basis of the cloud's server and network resources, which is widely known as utility computing. Such a service model may appear to overcome the effects of a DDoS attack, i.e., resource bottlenecks are eliminated. However, the cloud merely transforms a conventional DDoS attack on server and network resources to a new breed of attacks that target the cloud adopter's economic resources, originally labeled as Economic Denial of Sustainability (EDoS) attack by Hoff [2]. Therefore, unlike conventional DDoS attacks, an EDoS attack ultimately targets the financial resources of an organization, but not its physical network or server resources.

An EDoS attack occurs when zombie machines (part of a botnet) send a large amount of undesired traffic toward the cloud, exploiting the cloud's scalability, to chalk up an exorbitant amount of cost on a cloud adopter's bill. In other words, the attack is making the cloud unsustainable by fading the cloud billing mechanism to charge the cloud user's bill for the attack's activities. For example, a company taps into Amazon EC2 or Google App Engine for their application, and pays for the computing and bandwidth costs based on usage. Depending on the traffic, the service usage can be scaled up or down. Attackers could send malformed requests blended with legitimate ones to the applications running on these services. Since these requests are consuming the computing resources including inbound and outbound bandwidth traffic, and the CPU cycles for processing such requests, the applications vendor's cloud bill gets charged an excessive amount of expenditure. Unless the application vendor or the cloud providers have a smart technique to stop such risk, a sustained attack like this could ramp up the applications vendor's cloud infrastructure bill.

In this paper, we present an analytical model to study the impact of EDoS attacks on a single-class cloud services in which there is only one type of application service provided in the datacenter. The model considers a number of performance metrics. These metrics include end-to-end response time, utilization of computing resources being consumed, and the incurred cost resulting from the attack. Such model is convenient to show the impact of an EDoS attack on both performance and cost of the cloud computing services. Although we concentrate on modeling the EDoS attack against cloud computing, the proposed model is also suitable for other similarly behaving attacks such as DDoS attacks discussed by Zlomislic et al. [3].

The rest of the paper is organized as follows. Sections 2, 3, and 4 present the proposed architecture, analytical modeling, and simulation model for the cloud service under an EDoS attack, respectively. Section 5 presents the results and discussions. In Sect. 6, we discuss the related works. Finally, the conclusion and the future work are presented in Sect. 7.

2 Proposed Architecture of Cloud Web Service

Our study, in this paper, focuses on the evaluation of the EDoS attack on a cloud service of Software as a Service type (SaaS) such as a web application service. This could be considered as a single-class service in which there is only one kind of application service provided in the datacenter. The attack utilizes the scalability nature of the cloud to charge the cloud adopters an extra cost for the attack activities. For the attack to have higher malicious influence on the cloud economics, it is required to flood the cloud service by heavy workload. The goal is to force the cloud provision technique to add more instances to manage this workload and satisfy the Service Level Agreement (SLA) requirements for the target cloud service.

Figure 1 shows a cloud-based web service architecture drawn based on the given specifications and architecture of most cloud computing providers like Amazon Web Application Hosting [4]. The main components are the Load Balancer (LB) service, Virtual Machine (VM) instances, and the storage service.

The LB passes the clients' requests through to a pool of available VM instances that represent the web/application service [5]. VM instances are clustered in elastic groups to which users associate triggers. These triggers will automatically scale VM resources based on bandwidth or CPU utilization measured by a monitoring system such as Amazon CloudWatch web service. The LB ensures an even distribution of the incoming load among all running VM instances in a group [6,7].

VM instances run simultaneously as web application service centers, each potentially having a queue to process client requests [8]. The scalability of the service can be controlled by varying automatically the size of the group based on para-

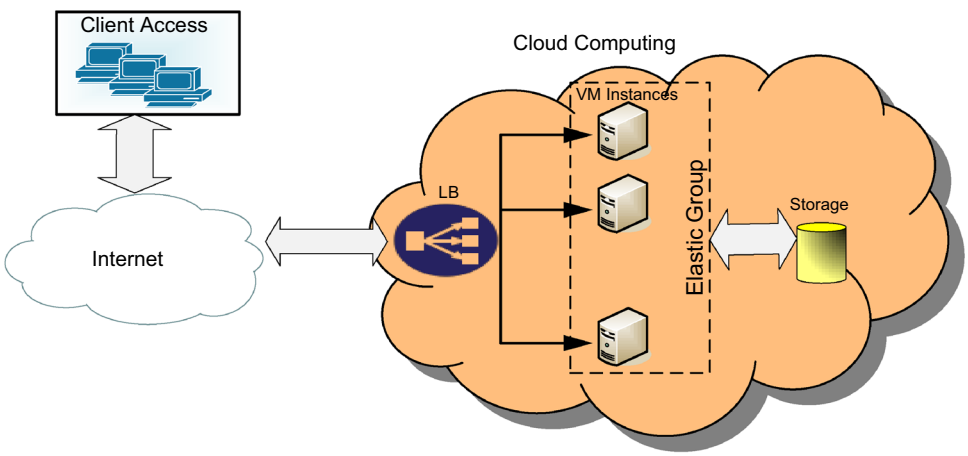


Fig. 1 Cloud-hosted scalable web service

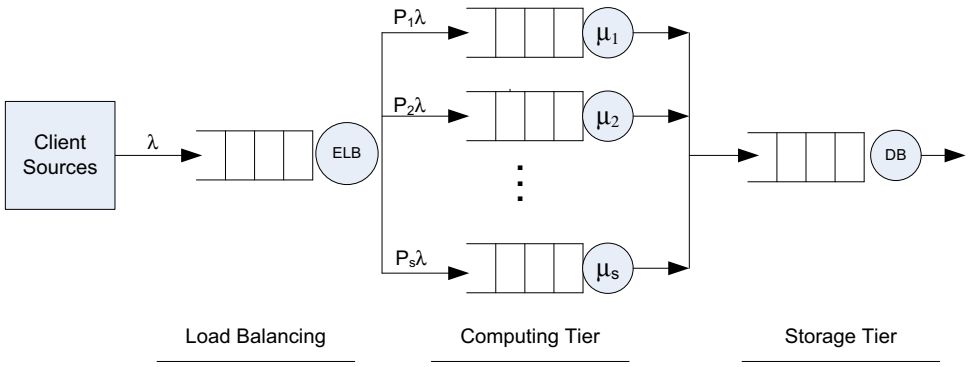


Fig. 2 Queuing model for the cloud-based web application

meters such as the average CPU utilization of the running instances [9]. For example, when the average CPU utilization for a group exceeds an upper threshold, a trigger is fired to create a new instance that will be attached to the group and registered at the LB.

Like most web application architectures, a cloud-based web application service has a database server to be used for caching and storing configuration information [10]. For example, a Relational Database Service (RDS) could be used to enable a web application caching tier.

For most of the web applications, the number of client requests and the service rate are considered to be random variables having Poisson distribution [11–14]. Similarly, it is an adequate and reasonable assumption for a cloud-based web application to consider that the requests arrival rate and their service rate follow a Poisson distribution. Several studies have assumed exponential distribution for both the requests inter-arrival time and the requests service time in a cloud service [15–18]. In addition, we are focusing on the EDoS attack targeting a single-class service where all cloud customers' requests have the same processing procedure as it is in a web service that delivers content, such as web pages, using Hypertext Transfer Protocol (HTTP) over the Internet. Thus, considering a Poisson distribution for the service rate in our case is a valid assumption that also helps in simplifying the proposed performance model.

The cloud-based web application architecture shown in Fig. 1 can be reasonably approximated using an open queuing network. Figure 2 shows an open queuing network model that mimics the required cloud-based web application stages including the load balancing service, the computing tier represented by a group of parallel VM instances, and the cloud storage tier.

The LB is modeled as $M/M/1$ queuing model, considering the use of a randomized algorithm to balance the load among the available instances so that each instance has an equal probability of receiving a request [8].

VM instances are modeled as parallel queuing models each as $M/M/1$. It is worth noting that, in reality, a cloud instance has a bounded buffer queue, i.e., $M/M/1/k$, but for approximation and convenience, we use $M/M/1$. Such approximation is highly accurate for systems with large finite buffers, such as cloud computing systems, where the probability of overflow of the buffer is negligible [19,20].

Modeling each compute instance as M/M/1 corresponds to the architecture of the instance as it has its own network interface (NIC), computing resources (CPUs), memory, and storage [8,21]. For example, with using Single Root I/O Virtualization (SRIOV)-capable Network Interface Card (NIC) [22], a VM could be bound to a Virtual Function (VF) driver that provides an abstraction of a dedicated NIC. Thus, when a packet is routed to the VM, it will be copied to the local queue assigned to the VM by the VF that gets executed with the virtual interrupt controlled by the SRIOV [23]. In addition, the LB is considered to be efficient and fast; otherwise, it will be a bottleneck in the cloud as it represents the public access point to the cloud services. Thus, the other side (receiver), which is the computing instance, should have a queue to hold the arriving requests. Moreover, several studies have modeled a web service as a network of queues, in which each machine in the distributed system is modeled as a single queue [14,24,25]. Other studies have also proposed modeling each VM as a single queue for different purposes [8,16,18,26,27].

The storage tier can be implemented using RDS, such as Amazon RDS which provides a managed relational database in the cloud as a web service. RDS offers several capabilities such as scaling up the compute and storage resources, monitoring the database health, point-in-time recovery for the database instance, and managing automated backups. Accordingly, RDS is considered to be vertically scalable on the database tier, and it is not able to scale-out by adding database servers due to its classic architecture [28]. Thus, the cloud storage service with such characteristics could be modeled as a single M/M/1 queue as it also has been discussed in [21,29].

However, since the EDoS attack mainly utilizes the scalability of the computing resources to maliciously charge the user, we concentrate on modeling the computing layer of the cloud service to analyze the impact of such attacks on the cost and performance of the targeted cloud service. Furthermore, the utilization of cloud storage can be ignored by assuming

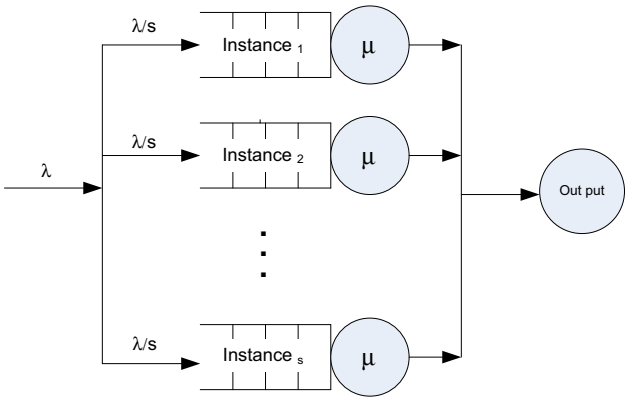


Fig. 3 Queuing model for the computing resources

that all data requirements for the execution of a request are met by the virtual web application instance that has its networking, computing, and storage resources to process the requests [8,16]. Thus, the problem can be reduced to the queuing model presented in Fig. 3 as an open queuing network model.

Since the LB ensures an even distribution of the traffic among the instances, the transition probability shown in Fig. 2, P_i, will be equal to $1/S$ for all the computing instances, where S is the number of running instances in the auto-scaling group. Assuming that all the computing instances have the same computing power capacity, $\mu_i = \mu$, and the arrival rate at each instance is $\lambda_i = \lambda/S$, the equations of such model will be as follows.

The mean computing utilization U is calculated as follows:

$$U = \frac{\sum_i^S \frac{\lambda_i}{S\mu}}{S} = \frac{\lambda}{S\mu} \tag{1}$$

The mean response time, based on the given open queuing network model of S parallel single queues [30], can be calculated as described below.

Since the LB is presumed to evenly distribute the requests among the S running instances, the routing matrix will have probabilities, Pi, that are equal to $1/S$, where Pi is the routing probability to the ith instance. As a result, the total input into each instance is:

$$\Lambda_i = \frac{\lambda}{S}$$

The average delay of a request in instance i can be calculated based on $M/M/1$ queuing theory to be:

$$\bar{T} = \frac{1}{\mu - \Lambda_i}$$

Using Little's formula [31], the average number of requests in instance i is $\Lambda_i \bar{T}_i$. Thus, the total number of requests in the network is:

$$\bar{N} = \sum_{i=1}^{S} \Lambda_i \bar{T}_i = \sum_{i=1}^{S} \frac{\Lambda_i}{\mu - \Lambda_i}$$

In addition, the total rate of the request flow into the network is $\sum_{i=1}^{S} \Lambda_i = \lambda$. Thus, by applying Little's formula, the average delay of the network is:

$$\bar{T} = \frac{1}{\lambda} \sum_{i=1}^{S} \frac{\Lambda_i}{\mu - \Lambda_i}$$

Assuming that auto-scaling of cloud instances is enabled and there is no delay in binding new instances to the group, the average response time of a request in the network, Rt will be:

$$Rt = \frac{S}{S\mu - \lambda} \tag{2}$$

It should be noted that when comparing the obtained equations for the given queuing model to $M/M/1$ model with service rate of $S\mu$ both have the same mean computing utilization, U.

The mean response time, considering a single queuing model [30], is as follows:

$$Rt_{mm1} = \frac{S}{S\mu - \lambda}$$

3 Analytical Modeling of the EDoS Attack

Figure 4 shows the proposed queuing model for capturing the cloud service considering an EDoS attack with a rate of λ_m, and legitimate traffic with a rate of λ_l requests per second.

The assumption of Poisson arrival for the DDoS attack has been discussed in [32–35]. Since the EDoS behaves similarly to DDoS in generating malicious flooding traffic, we have assumed Poisson traffic for the EDoS attack. Although attackers can choose any distribution to generate the traffic, the more attractive one is the distribution that is closer to the behavior of the legitimate traffic.

According to Poisson composition property [30], the aggregated traffic from multi sources each having an arrival of Poisson process follows a Poisson process with an average arrival rate of $\lambda = \lambda_i + \lambda_m$. Thus, each node in the proposed queuing model has a Poisson arrival.

The intended metrics for the proposed model can be calculated as described below.

The mean utilization of the computing resources of the running instances can be calculated as:

$$U = \frac{\lambda_l + \lambda_m}{S\mu} \tag{3}$$

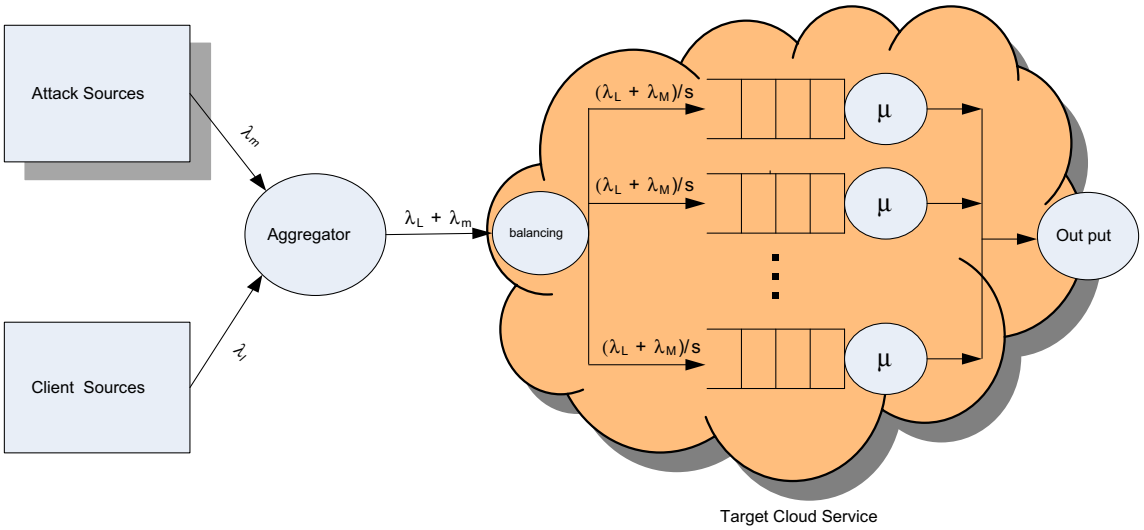


Fig. 4 Queuing model for an EDoS attack against a cloud service

The utilization incurred by the attack is:

$$U_m = \frac{\lambda_m}{S\mu}$$

It has an impact on both cost and performance metrics of the offered cloud services.

The average response time *Rt*, can be calculated based on Eq. (2) to be:

$$Rt = \frac{S}{S\mu - (\lambda_l + \lambda_m)} \tag{4}$$

For the throughput, it can be calculated directly from Little's formula. When the number of running instances at the cloud service side is S, the arrival rate $\lambda = \lambda_l + \lambda_m$ will be distributed among S instances due to load balancing. Thus, each instance will have $(\lambda_l + \lambda_m)/S$ as its arrival rate. According to Little's formula, the throughput of an M/M/1 queuing system, with arrival rate of $(\lambda_l + \lambda_m)/S$ and a service rate of μ, is $\mu \times \rho = \mu \times ((\lambda_l + \lambda_m)/S\mu) = (\lambda_l + \lambda_m)/S$, where $\rho \leq 1$. As a result, the average throughput at the cloud service with S running instances will be $\lambda_l + \lambda_m$.

One of the measurements that we have studied, and which is the target of an EDoS attack, is the cost associated with both the computing resources and the bandwidth on the cloud service side.

There are several pricing models that could be adapted to the cloud computing system. Currently, Amazon mainly offers computing instances with three pricing models including on-demand, spot pricing, and fixed pricing models [36]. The on-demand model allows the cloud user to pay for the used resources by the hour with no long-term commitments. In the fixed pricing model, a cloud user reserves cloud instances with one-time payment for each instance, e.g., for a 1 or 3 year period; and in turn receives a significant discounted hourly pricing on usage. The spot pricing model, offered by Amazon EC2, allows a cloud user to bid for available EC2 capacity and grants the user the requested resource only if the user's bid price is above the current spot instance price, which fluctuates periodically based on supply and demand.

A cloud user has to pay for the computing resources, the network traffic volume, and for the storage service, if required. In our work, we follow the on-demand pricing model considering only the cost related to both the computing usage and bandwidth usage.

The cost with regard to the computing resources has been calculated based on as follows:

$$COST_{\text{com}} = \sum_{i}^{n} \Pr ice_{\text{com}} \times t_i \times S_i \tag{5}$$

where $Price_{\text{com}}$ is the base price charged for the amount of computing resources per hour per instance, and S_i is the number of running instances during the period t_i.

Since we are interested in calculating the cost incurred by the attack, assuming it lasts for T hours, the total cost for the duration of the attack can be expressed as follows:

$$COST_{\text{com}} = \Pr ice_{\text{com}} \times T \times S \tag{6}$$

The cost related to the bandwidth can be calculated as follows:

$$COST_{bw} = \Pr ice_{bw} \times \bar{\lambda} \times T$$

where $\bar{\lambda}$ is the effective arrival rate measured in GB/s, and $Price_{bw}$ is the price per GB. When assuming that the queuing-based loss probability of cloud service is zero, the

effective arrival rate $\bar{\lambda}$ equals the arrival rate λ. Thus, the total cost can be expressed as follows:

$$COST = \left(\Pr ice_{bw} \times \lambda_{GB/s} + \Pr ice_{\text{com}} \times S\right) \times T \quad (7)$$

where $\lambda_{GB/s}$ is the effective arrival rate in GB/s, and is equal to $\lambda_l + \lambda_m$.

The number of instances committed to the cloud application service could be calculated based on Eq. (3), when assuming 100 % as the upper threshold utilization for provisioning, to be as follows:

$$S_{required} = \left\lfloor \frac{\lambda_l + \lambda_m}{\mu} + 1 \right\rfloor$$

Thus, the number of instances is changed based on the arrival rate of requests and the capacity of the cloud instance.

One of the main characteristics of the cloud computing is its elasticity which allows for the resources to be scaled up or down based on some monitored metrics of the cloud computing services. Statistics about these metrics are collected during a window time called the monitoring window. When a particular metric indicates a value above a given upper threshold, a policy is triggered to provision more resources so as to enhance its performance with respect to this metric. On the other hand, when a metric has a value below a given lower threshold, a policy is triggered to terminate some resources. This will enhance the usage of resources and keep the monitored metric value above the given lower threshold.

According to the auto-scaling service offered by Amazon [9], thresholds are used to trigger the provision mechanism to increase or decrease the number of running instances committed to an auto-scaling group. When the mean of CPU utilization, U, is above the upper threshold, *Upper*, the number of instances should be increased either by a specific number of instances or by a percentage of the current used resources to handle an increase in traffic. Similarly, when U is below the lower threshold, *Lower*, the number of instances should be decreased to more efficiently use the committed computing resources. For example, when we consider adding or removing instances by 10 % of the running instances at the time of the threshold triggering, the provisioning can be expressed as follows:

$$\begin{aligned} \text{When } U > Upper, \quad & S_{new} = [S + \lceil 0.1 \times S \rceil] \\ U < Lower, \quad & S_{new} = [S - \lceil 0.1 \times S \rceil] \end{aligned}$$

where S is the current number of running instances, and S_{new} is the updated number of instances after the occurrence of the trigger.

The optimal number of instances required to process an observed load of λ can be calculated as follows:

$$\frac{\lambda}{S\mu} \leq Upper\_threshold$$

Thus,

$$S = \left\lceil Upper\_threshold^{-1} \times \lambda/\mu + 1 \right\rceil \quad (8)$$

where λ and μ represent the arrival rate into the auto-scaling group and the service rate of one instance, respectively.

In the case of 80 % as the upper utilization threshold, the number of required instances can be calculated using Eq. (8) as follows:

$$S = \lceil 1.25 \times \lambda/\mu + 1 \rceil$$

Thus, the optimal number of instances required to be added to cope with the spike of λ is:

$$needed\_res = 1.25 \times \lambda/\mu + 1 - run\_res,$$

where run_res is the number of already running instances.

The required instances will gradually be committed to the service by adding a specific number of instances every time the provisioning takes place.

However, there are many factors related to the resource provisioning in cloud computing that might affect the performance of the cloud services. These factors include the overhead when allocating an instance to the cloud service, thresholds used to control the provisioning event such as the utilization threshold, the number of instances added every time the provisioning takes place, and the monitoring window time used for collecting the statistics.

4 Simulation Model

We have conducted a discrete-event simulation experiment to evaluate the performance of the cloud service under the EDoS attack in terms of key performance indicators including end-to-end response time, computing resources utilization, and throughput. Since the EDoS attack is mainly targeting the cloud adopter, we have also evaluated the cost associated with the computing resources and bandwidth allocations at the cloud service side.

The simulation followed closely the guidelines given by Law and Kelton [37], including the use of initial seeds that were ten million apart, and avoiding any overlapping in the random number streams during the simulation. Proper seed selections have to be made in order to avoid wrong combinations of seeds and random number generators that may lead to erroneous results. A different stream is generated for each simulation variable. Here are briefly some of the guidelines that are followed in selecting seeds [37]:

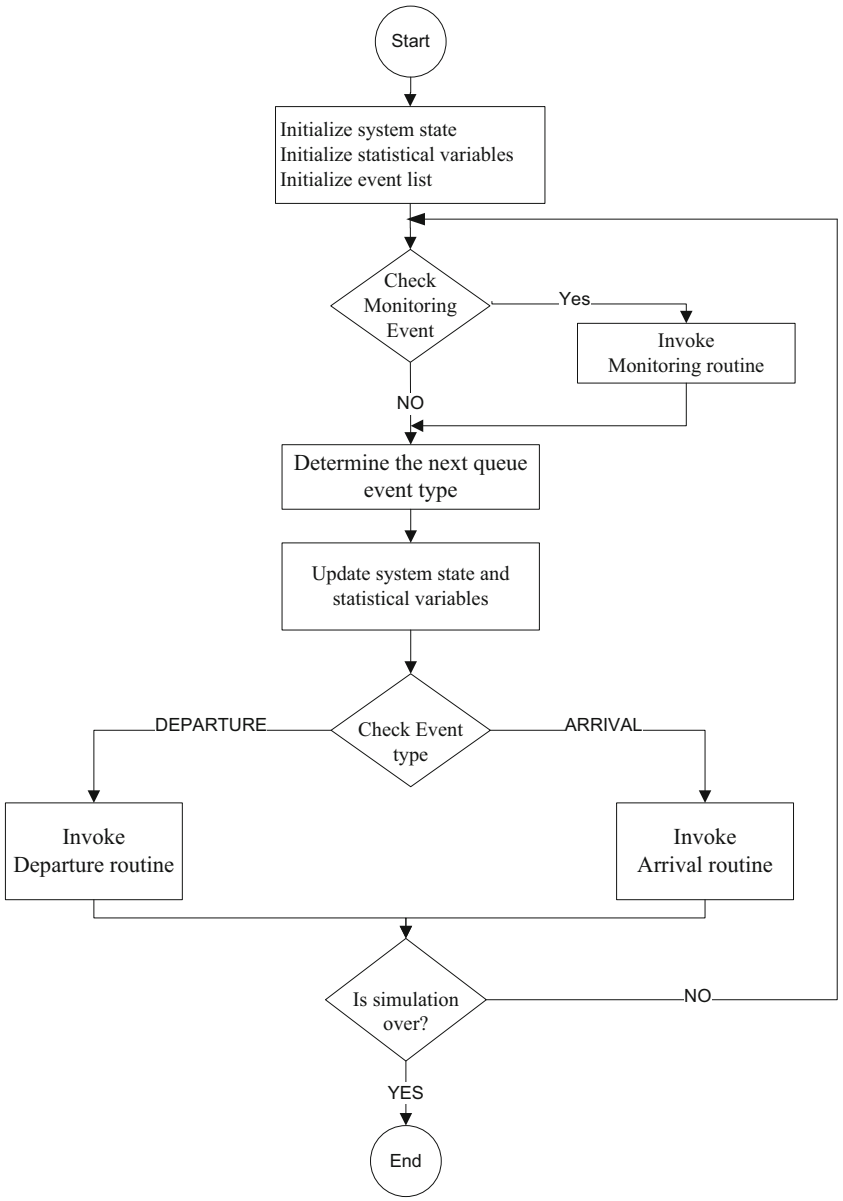
- Arbitrary values for seeds were not used. Also, the values of zero and even values were not used.
- Every simulation variable has its own stream, and streams were not subdivided.

- Overlapping of streams, to prevent correlation, was avoided by choosing seeds spaced 10,000,000 apart.

Figure 5 depicts the general flowchart of the simulation model that is applied for each queue in our simulation. Our simulation model has two types of events including the ARRIVAL and DEPARTURE events. The ARRIVAL event occurs when a new request arrives to the system. The DEPARTURE event occurs when a request is completely processed by the system. The two events are generated independently such that each event has its own seed and random number stream. Algorithm-1 and Algorithm-2 show the implementation of Arrival and Departure routines, respectively.

In addition, we have considered a Monitoring time event for the whole system. By the Monitoring event, we have simulated and controlled the provisioning mechanism of the cloud computing resources. Based on the cloud computing elasticity, the resources can be scaled up or down based on some monitored metrics of the cloud computing services. Statistics about these metrics are collected during a window time called the monitoring window. When a particular metric indicates a value above a given upper threshold, a policy is triggered to provide more resources. On the other hand, when a metric has a value below a given lower threshold, a policy is triggered to terminate some resources. Algorithm-3 shows the implementation of the Monitoring routine.

Fig. 5 Simulation flowchart



Algorithm-1: Arrival_routine

```
Begin
  Add request to the queue
  if (Instance_Status=IDLE)
    Begin
      Instance_Status ← BUSY
      Schedule the next DEPARTURE event
      Update the statistics
    End
  else
    Begin
      if (Space available)
          Queued the request
      else
          Drop the request
    End
  Schedule the next ARRIVAL event
End
```

Algorithm-2: Departure_routine

```
Begin
  Update the statistics
  Remove  request from the queue
  if (queue not empty)
    Begin
      Pick a request from the queue
      Schedule the next DEPARTURE event
      Update the statistics
    End
  else
    Instance_Status=IDLE
End
```

The simulation has been implemented using C language as it provides the best flexibility in coding the queue structure and events, random number generation, and other features necessary for simulation validation.

Algorithm-3: Monitoring routine

```
Begin
  Avg_Util ← Calculate average CPU utilization of the group
  if (Avg_Util > Upper-Threshold)
    Begin
      X← Determines number of extra instances
      Schedule the Add_Instances (X) event
    End
  else if (Avg_Util < Lower-Threshold)
    Begin
      Y← Determines number of instances to be terminated
      Schedule the Terminate_Instances (Y) event
    End
  Update the Monitoring_Time
  Schedule Monitoring-Event()
End
```

Each instance has been implemented as a FIFO queue that holds the packets arrival times currently in the system to be used for statistic gathering. The ARRIVAL and DEPARTURE events related to each instance have been implemented as it is described in Algorithms 1 and 2. A priority queue is used to invoke the next event based on a "time" value assigned from the random number generator.

We have followed the Inverse Transformation [38] method to generate random number varieties for the exponential distribution. The Inverse Transformation method uses uniform deviates, U(0,1), which are random numbers uniformly distributed between 0 and 1. For the uniform deviates, we have used PMMLCG (prime modulus multiplicative linear congruential generator) as recommended by [37].

Finally, the relative error is used to measure the accuracy of the queuing model results compared to the simulation model results. The percentage of the relative error is defined as follows:

$$\textit{Relative Error} = \left| \frac{\textit{Queuing Results} - \textit{Simulation Results}}{\textit{Simulation Results}} \right| \times 100 \tag{9}$$

5 Results and Discussion

We have considered two simulation scenarios using the simulation model discussed in the previous section. In the first scenario, we have considered different attack rates to show the impact of the attack on the targeted cloud service. The second scenario is for the optimal case where there is no attack targeting the cloud service. In addition, the output of the proposed analytical model has been compared to the simulation results.

In the simulation experiment, we have considered the same setup as that of the queuing model presented in Fig. 4. The input to the simulation is an aggregated traffic from different sources including attackers' traffic. We have considered the Poisson nature of the incoming traffic as was clarified in Sect. 2. We have assumed a fixed input rate of 400 Req/sec (request per second) representing the rate of the legitimate requests coming from clients and a variable input rate ranging from 400 to 8,000 Req/sec representing the rate of the attack traffic.

To determine the simulation run length, we first applied the Welch technique to eliminate the warm-up period [37]. Welch's technique is based on the plotting of moving averages calculated for the means of the observations made in replications. The warm-up period is selected at the point at which the plot becomes smooth. Then, the length of the simulation is set to five times the length of the warm-up period. The number of arrival events was found to be about 5 million to reach the steady state.

We assume a high load spike similar to the ones caused by DDoS attacks. Thus, we have only considered the upper threshold for adding more instances so as to cope with the load spikes. Such upper threshold has been discussed in previous works [7,16] to be 80 %. We considered the scaling size to be two instances per provisioning occurrence. The time window that we have used to monitor the resources utilization is 5 min, as it is the default period used in the Amazon auto-scaling mechanism [9].

We assumed using a small instance that has a capacity of 100 requests per second as it was discussed by Catteddu and Hogben [39]. The number of initial running instances is 5 instances which can handle 200 Req/sec assuming 50 % utilization. Furthermore, we considered the overhead caused by committing instances to the cloud service to be 55.4 s for provisioning one VM instance, as was measured by Islam et al. [40].

The cost has been calculated based on Eq. (7). The $Price_{com}$ has been set to \$0.115 as it is recently reported in Amazon for small on-demand instances running on the Windows operating system [36]. Regarding the cost associated with the bandwidth allocation, we have used a base price of \$0.01 per GB in/out data transferred based on the reported prices of Internet data transfer "in" and "out" of Amazon EC2 [36].

Figure 6 shows the obtained results regarding the end-to-end response time of the legitimate requests. The results show that when the load increases, the corresponding response time also increases. It is obvious that the response time does not go up considerably when the attack traffic increases to very high values. This is due to the auto-scaling mechanism that allocates more instances to process the high load caused by the attack traffic. However, the results show that, in general, the attack makes the legitimate clients suffer more response time compared to the optimal case.

Figure 7 shows the evaluation of the computing resources utilization. Results show a trend similar to the one of the end-to-end response time in Fig. 6 such that as the attack rate increases, the utilization increases. It is obvious from the results that the average utilization does not exceed the upper threshold, 80 %, used in both simulation and analytical models. Moreover, the results show that the EDoS attack consumes more computing resources when compared to the optimal case where there is no attack. For instance, based on the numerical results obtained from the simulation, at an attack rate of 6 KReq/sec, the mean utilization for 79 running instances is about 79 %, whereas for the normal case, the mean utilization is only about 67 % while using only 6 instances.

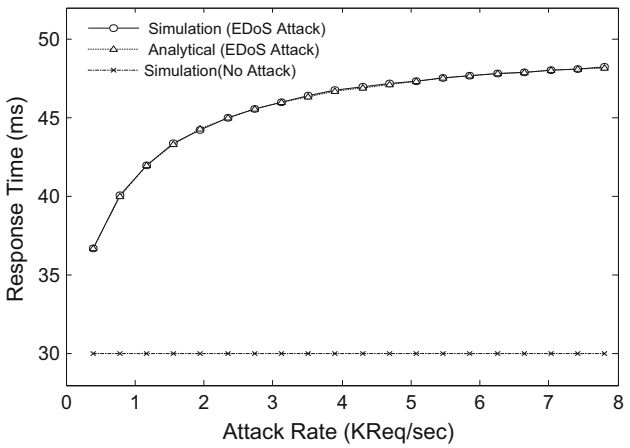


Fig. 6 Response time in relation of attack rate

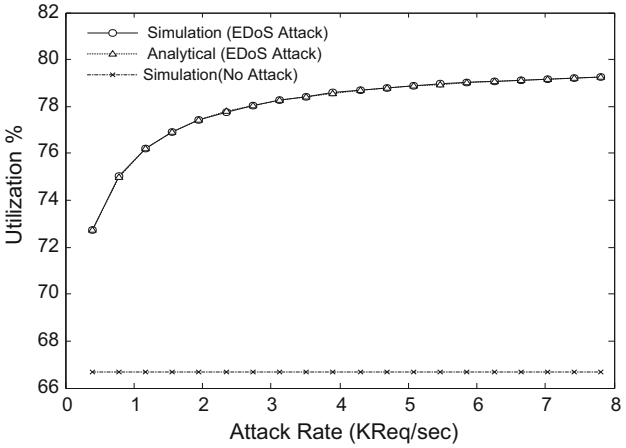


Fig. 7 Compute utilization in relation to attack rate

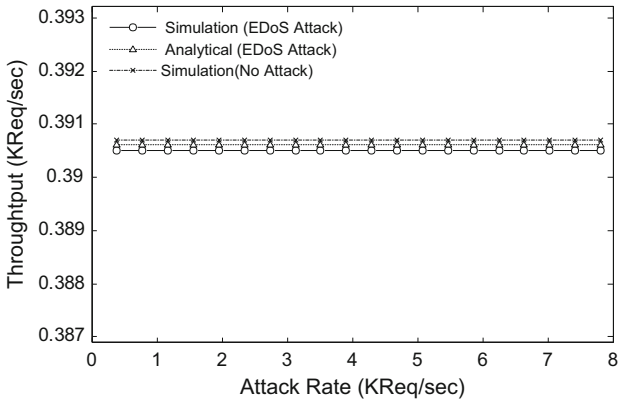


Fig. 8 Throughput of legitimate requests in relation to attack rate

Regarding the throughput rate, it is expected that the throughput of the legitimate requests will not be affected by the attack rate due to the fact that the targeted cloud service is an on-demand cloud-based. According to scalability nature of the cloud computing system, we are assuming that there are enough on-demand cloud resources to be provisioned to the cloud instances executing the service. As a result, there is no or little noticeable degradation of the throughput rate of the legitimate requests. Figure 8 shows the same expected trend of the throughput of the legitimate requests.

As for the cost evaluation, Fig. 9 shows an increase in the cost when the attack rate increases. In fact, the extra cost added because of the EDoS attack is very high when compared to the optimal cost where no attack takes place. For instance, at an attack rate of 6 KReq/sec, the total cost is about 15 times the normal one.

Figure 10 shows the resulted relative error percentage for the response time, utilization, throughput, and cost results

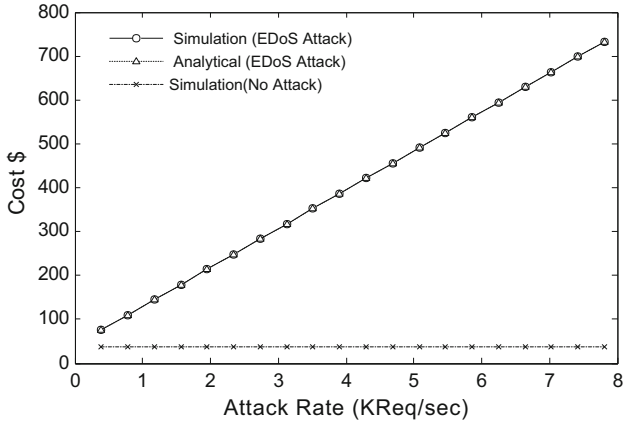


Fig. 9 Incurred cost in relation to attack rate

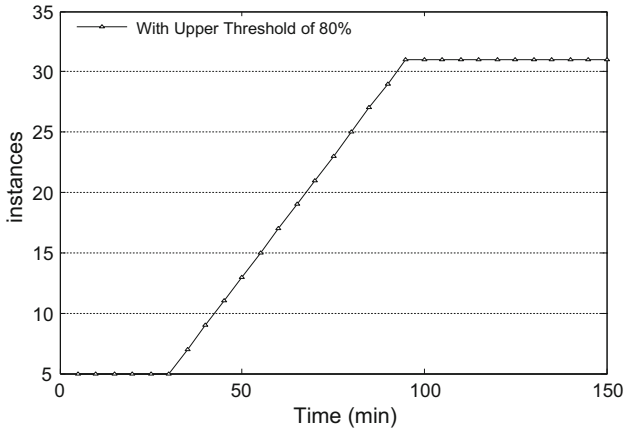


Fig. 11 Simulation results of the number of allocated instances

when comparing the queuing model to the simulation model. It shows a good accuracy for all the studied metrics with maximum error of about 0.1 %.

As a verification of the simulation results, we have conducted a simulation experiment to show the number of allocated instances along the simulation time. In this simulation run, the arrival requests rate was assumed to be 200 Req/sec during the early periods of the simulation and then it increases to be 2,400 Req/sec after 25 min. The number of initial running instances is 5 instances which can handle 200 Req/sec. This means that the load was increased 12 times indicating a high load peak.

Figure 11 shows the results of the number of instances allocated during the simulation run. For the load of 2,400 Req/sec, results show that 31 instance is the minimum number of instances required to insure that the average utilization is below 80 %. According to Eq. (8), the number of required instances can be calculated analytically as: $S = \lceil 1.25 \times 2{,}400/100 + 1 \rceil = 31$, which comes in line with the simulation results.

6 Related Work

There are several proposed works in the literature to model a service exposed through cloud computing. Xiong et al. [41] have proposed a queuing theory-based method for studying the performance of a cloud service in terms of the response time. In their model, the cloud computing service has been modeled as a tandem of two $M/M/1$ queues representing the web server and the service center for single-class customers.

Chen and Li [11] have proposed a queuing-based model for dynamically provisioning cloud computing virtual machines to meet the service level agreement (SLA). They have proposed using $M/M/S/k$ queuing model to capture the architecture of the web application in the cloud computing. In their work, they have considered the web application as a centralized queue and the virtual machines as service centers with finite caches and buffers. Similarly, Hu et al. [42] have proposed modeling computing resources in the cloud as a multi-server queuing model, $M/M/S$. The purpose of such model is to guide resource allocation decisions in terms of the minimum number of servers that should be allocated to each application environment to meet the SLA.

Bi et al. [26] have proposed a dynamic provisioning model for virtualized multi-tier applications in cloud data centers based on open queuing networks. The virtualized multi-tier application in cloud computing is deployed on multiple VMs for each tier that provides certain functionality to its preceding tier (e.g., web server, database). They constructed a hybrid model that represents the scheduling tier as an $M/M/s$

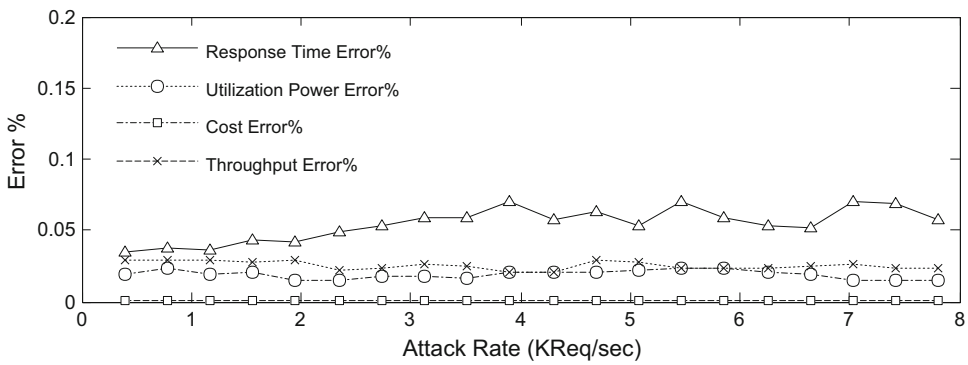


Fig. 10 Relative error percentage when comparing simulation and analysis results

queuing system, and multiple $M/M/1$ queuing systems for the other tiers.

Shi et al. [18] have developed efficient energy saving methods in the cloud datacenter by dynamically allocating resources based on utilization analysis and prediction. The main prediction scheme that has been used in their work was an $M/M/1$ queuing model that captures the cloud-based web service. Similarly, Calheiros et al. [16] have proposed an adaptive provisioning technique for cloud-based resources to deliver cloud-based applications that meet QoS targets based on queuing network system model and workload information. They model each virtualized application instance as an $M/M/1/k$ queuing model, where k refers to a finite queue of length k.

However, we believe that the structure of a cloud is similar to multiple queues rather than a centralized queue with multiple servers; since each cloud instance has its own network interface, computing resources, memory, and storage [8]. Several studies have modeled a web service as a network of queues, in which each machine in the distributed system is modeled as a single queue [14,24–26]. A VM in the cloud computing system can be viewed as a machine in a distributed system offering a web service, and which could be modeled as a single queue [21]. Thus, modeling each cloud instance as an $M/M/1$ is closer to the real-world deployments.

In addition, according to the elasticity of the cloud computing system, a cloud-based service usually has multi-cloud instances (like EC2) offering the service to the cloud users. Thus, modeling a cloud service by using a multi $M/M/1$ queuing model is closer to the reality.

Sqalli et al. [43] and Al-Haidari et al. [44] have proposed mitigation techniques to overcome the impact of an EDoS attack on cloud computing. The proposed techniques depend on the collaboration between a firewall and a verifier node for the purpose of detecting and then dropping the malicious requests before reaching the protected server. However, the focus in our prior works [43,44] was on studying the performance and the capabilities of the proposed mitigation techniques rather than studying the impact of the EDoS attack.

In contrast to previously published work, our paper has the following distinct contributions. First, the paper introduces an analytical model to study the impact of the EDoS attack on the cloud computing services. Our analytical model makes it easy to understand the behavior of the EDoS attack and its impact on the cloud computing services, since we have derived key important parameters of both the cloud computing service and the EDoS attack. Moreover, obtained results from the analytical model have been verified by the simulation model that mimics the cloud computing environment and the attack behavior. Second, in this paper, we have considered more detailed and realistic parameters for the simulation model such as the parameters of the provisioning mechanism (provisioning overhead, monitoring window time, and scaling factors) and the characteristics of the cloud computing services (realistic instance capacity and load balancing service). All of such parameters have been obtained from the Amazon cloud computing environment which makes the study more realistic. Third, we have studied and analyzed the cost metric, which has been ignored in the previous discussed works as they have only focused on the performance of the cloud computing system. In our work, we have calculated the cost based on queuing theory and using the on-demand pricing model offered by Amazon AWS Cloud service. Finally, the models will help in analyzing and studying the mitigation techniques against the EDoS attacks.

7 Conclusion

Security of cloud computing has been identified as a major concern and challenge to the adoption of this emerging technology. The paper presented a study about the impact of the EDoS attacks on the cloud computing services, considering only single-class services. We developed an analytical model verified by a simulation model to study such impact of EDoS attacks, considering a number of performance metrics. These metrics include end-to-end response time, utilization of computing resources being consumed, throughput, and the incurred cost resulting from the attack. The obtained simulation results are found to be in agreement with the analytical results, with a maximum relative error of about 0.1 %. Based on the obtained results, we found that the EDoS attack has a considerable impact on both the performance and the cost of the cloud services. For instance, results showed that at an attack rate of 6,000 Req/sec, the total cost can be as high as 15 times more than the normal usage with no attack. In addition to charging the cloud adopters more cost, EDoS attacks have an impact on the performance of the cloud computing services such as the end-to-end response time whereby unacceptable delays can be incurred. In addition, results have shown that there was little or no noticeable impact of the attack on the throughput of the legitimate requests as it is expected due to the scalability and availability of the cloud services. As a future work, we propose to study the impact of the EDoS attack while considering different pricing models such as the fixed and spot pricing models. In addition, we propose to study the impact of such attack on the cloud computing using an experimental test-bed.

Acknowledgments The authors would like to acknowledge the support provided by the 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 (NSTIP).

References

1. Gartner, Gartner Identifies the Top 10 Strategic Technologies for 2013. Analysts Examine Latest Industry Trends During Gartner Symposium/ITxpo, Orlando (2012)
2. Hoff, C.: Cloud computing security: from DDoS (Distributed Denial Of Service) to EDoS (Economic Denial of Sustainability). Blog. http://rationalsecurity.typepad.com/blog/2008/11/cloud-computing-security-from-ddos-distributed-denial-of-service-to-edos-economic-denial-of-sustaina.html. Retrieved 27 Nov 2008
3. Zlomislic, V.; Fertalj, K.; Sruk, V.: Denial of service attacks: an overview. In: 9th Iberian Conference on Information Systems and Technologies (CISTI), Barcelona, pp. 1–6 (2014)
4. AWS Documentation, AWS Web Application Hosting for Microsoft Windows. http://docs.amazonwebservices.com/gettingstarted/latest/wah/web-app-hosting-intro.html?r=1052
5. Amazon, Amazon Load Balancer Service. http://aws.amazon.com/elasticloadbalancing/
6. Buyya, R.; Ranjan, R.; Calheiros, R.N.: InterCloud: utility-oriented federation of cloud computing environments for scaling of application services. In: The 10th International Conference on Algorithms and Architectures for Parallel Processing, Busan, Korea (2010)
7. Bellenger, D.; Bertram, J.; Budina, A.; Koschel, A.; et al.: Scaling in cloud environments. In: Proceedings of the 15th WSEAS International Conference on Computers, Wisconsin, pp. 145–150 (2011)
8. Idziorek, J.: Discrete event simulation model for analysis of horizontal scaling in the cloud computing model. In: Proceedings of the 2010 Winter Simulation Conference, pp. 3004–3014 (2010)
9. Amazon Auto Scaling Developer Guide. Amazon Web Services LLC (2012)
10. Web application hosting in the AWS cloud: best practices. Amazon Web Services LLC (2010)
11. Chen, H.; Li, S.: A queueing-based model for performance management on cloud. In: 6th International Conference on Advanced Information Management and Service (IMS), Seoul, pp. 83–88 (2011)
12. Arlitt, M.; Williamson, C.: Internet web servers: workload characterization and performance implications. IEEE/ACM Trans. Netw. **5**(5), 815–826 (1997)
13. Walraevens, J.; Wittevrongel, S.; Bruneel, H.: Performance analysis of a priority queue with session-based arrivals and its application to E-commerce web servers. Int. J. Adv. Internet Technol. **2**(1), 46–57 (2009)
14. Liu, Z.; Niclausse, N.; Jalpa, C.: Traffic model and performance evaluation of web servers. Perform. Eval. **46**(2–3), 77–100 (2001)
15. Nan, X.; He, Y.; Guan, L.: Optimal resource allocation for multimedia cloud based on queuing model. In: IEEE MMSP, pp. 1–6 (2010)
16. Calheiros, R.; Ranjan, R.; Buyya, R.: Virtual machine provisioning based on analytical performance and QoS in cloud computing environments. In: International Conference on Parallel Processing (ICPP), Taipei City, pp. 295–304 (2011)
17. Pal, R.; Hui, P.: Economic models for cloud service markets. Lecture Notes in Computer Science, Distributed Computing and Networking,vol. 7129, pp. 382–396. Springer (2012)
18. Shi, Y.; Jiang, X.; Ye K.: An energy-efficient scheme for cloud resource provisioning based on cloudSim. In: 2011 IEEE International Conference on Cluster Computing (CLUSTER), Austin, TX, pp. 595–599 (2011)
19. Scheinhardt, W.: Markov-modulated and feedback fluid queues. Ph.D. Thesis, University of Twente, the Netherlands. http://www.ub.utwente.nl/webdocs/tw/1/t0000008.pdf (1998)
20. Shen, X.; Chen, H.; Dai, J.; Dai, W.: The finite element method for computing the stationary distribution of an SRBM in a hypercube with applications to finite buffer queueing networks. Queueing Syst. **42**(1), 33–62 (2002)
21. Dawoud, W.; Takouna, I.; Meinel, C.: Elastic VM for rapid and optimum virtualized resources' allocation. In: 5th International DMTF Academic Alliance Workshop on Systems and Virtualization Management (SVM), Paris, pp. 1–4 (2011)
22. Intel 82599 10 gigabit Ethernet controller. Intel (2009). http://download.intel.com/design/network/prodbrf/321731.pdf
23. Dong, Y.; Yang, X.; LI, X.; Tian, K.; Guan, H.: High performance network virtualization with SR-IOV. In: IEEE International Symposium on High Performance Computer Architecture (HPCA) (2010)
24. Sutton, C.; Jordan, M.I.: Bayesian inference for queueing networks and modeling of internet services. Inst. Math. Stat. Ann. Appl. Stat. **5**(1), 254–282 (2011)
25. Do, T.; Krieger, U.R.; Chakka, R.: Performance modeling of an apache web server with a dynamic pool of service processes. Telecommun. Syst. **39**(2), 117–129 (2008)
26. Bi, J.; Zhu, Z.; Tian, R.; Wang, Q.: Dynamic provisioning modeling for virtualized multi-tier applications in clouddata center. In: Proceedings of IEEE 3rd International Conference on Cloud Computing (CLOUD 2010), pp. 370–377 (2010)
27. Singh, R.; et al.: Autonomic mix-aware provisioning for non-stationary data center workloads. In: Proceedings of the 7th International Conference on Autonomic Computing, USA (2010)
28. Kossmann, D.; Kraska, T.; Loesing, S.: An evaluation of alternative architectures for transaction processing in the cloud. In: Proceedings of International Conference on Management of Data (SIGMOD) (2010)
29. Kihl, M.; Cedersjö, G.; Robertsson, A.; Aspernäs, B.: Performance measurements and modeling of database servers. In: Sixth International Workshop on Feedback Control Implementation and Design in Computing Systems and Networks (FeBID 2011) (2011)
30. Gross, D.; Shortle, J.F.; Thompson, J.M.; Harris, C.M.: Fundamentals of Queuing Theory. Wiley, New York (2008)
31. Little, J.: A proof for the queuing formula: $L = \lambda W$. Oper. Res. **9**(3), 383–387 (1961)
32. Liu, H.: A new form of DOS attack in a cloud and its avoidance mechanism. In: Proceedings of the 2010 ACM Workshop on Cloud Computing Security Workshop, Chicago, pp. 65–76 (2010)
33. Singh, N.; Ghrera, S.P.; Chaudhuri, P.: Denial of service attack: analysis of network traffic anormaly using queuing theory. J. Comput. Sci. Eng. **1**(1), 48–54 (2010)
34. Wang, Y.; Lin, C.; Li, Q.; Fang, Y.: A queueing analysis for the denial of service (DoS) attacks. Comput. Netw. **51**, 3564–3573 (2007)
35. Boteanu, D.; Fernandez, J.M.; McHugh, J.; Mullins, J.: Queue management as a DoS counter-measure? In: Garay, J.A.; Lenstra, A.K.; Mambo, M.; Peralta, R. (eds.) ISC 2007. LNCS, vol. 4779, pp. 263–280. Springer, Heidelberg (2007)
36. Amazon EC2 Pricing. http://aws.amazon.com/ec2/pricing/
37. Law, A.; Kelton, W.: Simulation Modeling and Analysis, 3rd edn. McGraw-Hill, New York (2000)
38. Jain, R.: The Art of Computer Systems Performance Analysis. Wiley, New York (1991)
39. Catteddu, D., Hogben G.: Cloud computing: benefits, risks and recommendations for information security. Technical Report, European Network and Information Security Agency (2009)
40. Islam, S.; Lee, K.; Fekete, A.; Liu, A.: How a consumer can measure elasticity for cloud platforms. Technical Report, School of Information Technology, Univercity of Sydeny (2011)
41. Xiong, K.; Perros, H.: Service performance and analysis in cloud computing. In: SERVICES '09: Proceedings of the 2009 Congress on Services—I (2009)
42. Hu, Y.; Wong, J.; Iszlai, G.; Litoiu, M.: Resource provisioning for cloud computing. In: Proceedings of the 2009 Conference of the

Centerfor Advanced Studies on Collaborative Research (CASCON '09), ACM, pp. 101–111 (2009)
43. Sqalli, M.; Al-Haidari, F.; Salah, K.: EDoS-Shield—a two-steps mitigation technique against EDoS attacks in cloud computing. In: Fourth IEEE International Conference on Utility and Cloud Computing (UCC 2011), Victoria, NSW, pp. 49–56 (2012)
44. Al-Haidari, F.; Sqalli, M.H.; Salah, K.: Enhanced EDoS-shield for mitigating EDoS attacks originating from spoofed IP addresses. In: The 11th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Liverpool, United Kingdom, pp. 1167–1174, 25–27 June 2012 (2012)