Detecting TCP SYN Flood Attack in the Cloud

Raneel Kumar[1*], Sunil Lal[2], Alok Sharma[1,3]

[1] University of the South Pacific, Fiji.
[2] Massey University, New Zealand.
[3] Griffith University, Australia.

* Corresponding author. Tel.: +679 323 2839; email: raneel.kumar@usp.ac.fj
Manuscript submitted April 5, 2017; accepted May 20, 2017.
doi: 10.17706/jsw.12.7.493-506

Abstract: In this paper, an approach to protecting virtual machines (VMs) against TCP SYN flood attack in a cloud environment is proposed. An open source cloud platform Eucalyptus is deployed and experimentation is carried out on this setup. We investigate attacks emanating from one VM to another in a multi-tenancy cloud environment. Various scenarios of the attack are executed on a webserver VM. To detect such attacks from a cloud provider's perspective, a security mechanism involving a packet sniffer, feature extraction process, a classifier and an alerting component is proposed and implemented. We experiment with k-nearest neighbor and artificial neural network for classification of the attack. The dataset obtained from the attacks on the webserver VM is passed through the classifiers. The artificial neural network produced a F1 score of 1 with the test cases implying a 100% detection accuracy of the malicious attack traffic from legitimate traffic. The proposed security mechanism shows promising results in detecting TCP SYN flood attack behaviors in the cloud.

Key words: Eucalyptus cloud, denial of service attack, TCP SYN flood, artificial neural network, k-nearest neighbor.

1. Introduction

Cloud computing is an emerging technological advancement in providing information technology infrastructure, platform and software as services over the Internet. Cloud computing is steadily being adopted by organizations as private, public or hybrid clouds and recently there has been an emergence of open source cloud platforms available for deployment, development and research into cloud computing. Although, cloud computing offers advantages such as reduced service cost and better utilization of resources [1], some organizations are reluctant in moving their corporate services on public clouds considering the security of their data and services. For organizations to transition to clouds, it becomes important for cloud providers to assure significant level of security to the clients. Together with existing security mechanisms such as firewalls and intrusion detection systems, cloud providers can also have security mechanisms built into the architecture of the cloud in assuring high level of security to the clients [2].

1.1. Cloud Computing

Cloud computing has a service driven model in providing cloud resources to the users [1]. Ideally, a cloud provider leases resources to users as grouped services from three conceptual layers; infrastructure as a

service (IaaS), platform as a service (PaaS), and software as a service (SaaS).

Cloud computing adoption can be categorized as public, private and hybrid clouds. The use or deployment of each of these types is based on the needs pertaining to it. Public clouds basically reside outside of an organization's premises and is accessible via the Internet. Many of the established cloud services such as Amazon Web Services [3], Microsoft Azure [4] and Salesforce [5] reside as public clouds that are accessible to users all over the globe and the users are mainly charged based on the services used. Private clouds reside in an organization's premises. Such clouds are preferred by large organizations who are sensitive to storing critical organization data on the public cloud due to the privacy and security issues. Over the years, a number of large organizations are preferring to have private cloud computing infrastructure for the organization's internal use and an example is the United States Central Intelligence Agency's (CIA) private cloud which had been setup by Amazon [6]. Hybrid clouds allow organizations to use a mix of private and public clouds to provide services. Using this model, organizations can move their on-premises services to public clouds when needed.

1.2. Denial of Service Attacks

Denial of Service (DoS) and Distributed Denial of Service (DDoS) attacks are one of the common attacks on the Internet today. DoS attacks aim to exhaust a system's resources such that it compromises its ability to provide the intended service and thus rendering it unavailable. The Cisco 2014 Annual Security report ranks the effects of DoS attacks in the magnitude of high severity [7]. DoS attacks which mainly target websites can also paralyze Internet service providers. For instance, in August 2013, the Chinese government reported that the largest DDoS attack it had ever faced shut down the Internet in China for about four hours. Also in March, 2013, a 300Gbps DDoS attack known as DNS amplification attack was observed on the Spamhaus website hosted at CloudFlare where around 30,000 DNS open resolvers were utilized to target the website [8]. DoS attacks can be generally classified into three categories [9]:

1. Volume based DoS Attacks: These affect servers when a high volume of such traffic is directed towards it. Examples are ICMP flood and UDP flood attacks.

2. Protocol based DoS Attacks: These use specific Internet protocols to consume the server's resources. Examples of these are TCP SYN flood attack and Ping-of-death.

3. Application based DoS Attacks: these target the weakness of the application. This is also known as Application Layer attacks and examples of this are Slowloris attack and DNS amplification attack.

2. Related Work

To secure cloud environments against DoS attacks a number of security mechanisms and frameworks have been proposed. An interesting work by Shea and Liu [10] highlight the performance of VMs under DoS attacks where the researchers experimented on different virtualization technologies of Kernel-based Virtual Machine (KVM), Xen, OpenVZ and Vanilla under TCP SYN flood attacks. A comprehensive set of benchmarks for CPU, network, memory and file system performance were considered. Although the research was centered more on virtualization than on cloud computing, its finding are important for this research as virtualization is at the core of cloud platforms. Another external DoS attack scenario is investigated by Lonua *et al.* [11] on a Eucalyptus private cloud. To detect the attacks from outside, the authors proposed to have Snort [12] set up on each VM and the monitoring and alerting information is directed to a basic analysis and security system with its web analysis tool being integrated to the front-end of Eucalyptus.

Gupta *et al.* [1] had proposed a profile based Network Intrusion Prevention System (NIPS) for securing cloud environments. The open source cloud platform used by the researchers was OpenNebula [13]. Here the NIPS which is managed by a cloud administrator examines packets originating from and destined to virtual interfaces of separate VMs and monitors it against the VM's profile. An initial VM profile is created by

monitoring all traffic that passes to and from the VM. This traffic is compared against an attack signature database and using the attacks and normal behaviors of the traffic a profile is created. The profile can be updated later by the administrator. The ICMP flood and TCP SYN flood attacks had been studied by the researchers. An alert and response component relayed the necessary information to the administrator. The research had presented a novel framework design of an intrusion prevention system but minimal evaluation of the DoS attacks detection accuracy had been presented.

Modi *et al.* [14] present a comprehensive review of intrusion detection techniques for cloud environments. They highlight on the use of data mining and machine learning techniques for an anomaly based intrusion detection system such as artificial neural networks, fuzzy logic, associate rules, support vector machine, genetic algorithm and hybrids of these. The authors do state that accuracy of some of these techniques are low. Another important factor to consider using these techniques is the experimentation dataset that is used. A common dataset that has been heavily used by researchers is the KDD CUP 1999 intrusion detection dataset despite it having some discrepancies as reported in literature – discrepancies such as imbalance of dataset and the aging factor of the dataset itself [15]. Singh and Bansal [16] have used NSL KDD dataset - a subset of the KDD CUP 1999 dataset with artificial neural network classifiers. They used multilayer perceptron, radial base function, logistic regression, and voted perception and evaluated the performance of each.

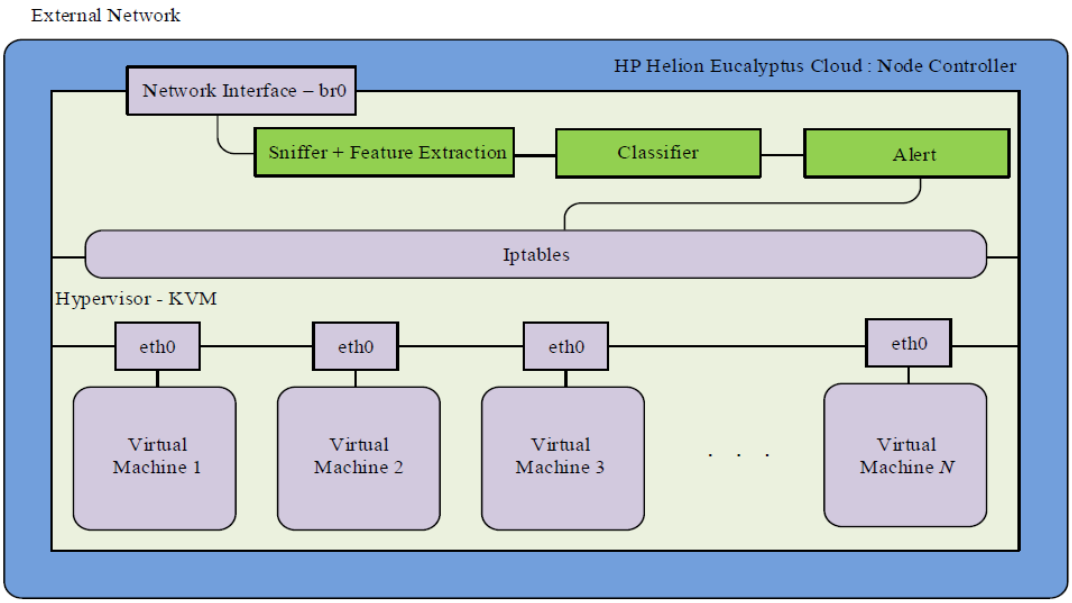

Fig. 1. Design of the DoS attack detection mechanism in Eucalyptus cloud.

3. Proposed Approach

As discussed in the literature, many researchers have worked on intrusion detection systems and mechanisms for DoS attacks on hosts residing on the cloud and the attacker is externally situated from the cloud. However, due to the multi-tenancy nature of cloud environments, especially for clouds deployed as IaaS models, out of all tenants residing on the cloud, there can be malicious tenants who can cause harm to legitimate tenants of the cloud. Thus, through this research work, the issue of DoS attacks within the cloud is explored. An open source cloud platform – Eucalyptus is used for the experimentation. DoS attack in the form TCP SYN flood attack is performed on a VM running a webserver. The webserver has the TCP SYN cookies enabled which is commonly considered to protect the servers from TCP SYN flood attacks [17].

The proposed approach (Fig. 1) is designed for the detection of TCP SYN flood attacks on VMs by VMs in

Eucalyptus cloud. The approach can be used for DoS attacks coming from the external network also, however in the experimentation all attacks are carried out by VMs within the cloud.

3.1. Eucalyptus Cloud Platform

Elastic Utility Computing Architecture for Linking Your Programs To Useful Systems (Eucalyptus) is an open source cloud platform under the Hewlett-Packard (HP) Helion [18] initiative which provides organizations opportunity to establish IaaS private clouds. Eucalyptus was released in 2008 and has over the years matured into a robust private cloud computing platform. Eucalyptus cloud offers many features such as compute, storage, networking, cloud management and compatibility with AWS APIs. Eucalyptus compromises of five main components and a sixth optional component of which all are stand-alone web-services. The components and the functions of each are provided in Table 1 [19]. For this research, experimentation has been carried out on Eucalyptus version 4.1.

Table1. Eucalyptus Components and Functionality

Components	Functionality
Cloud Controller (CLC)	Handle resource arbitration via the CC on the NCs
Walrus	Stores persistent data in the cloud such as machine images and snapshots
Cluster Controller (CC)	Receive requests from the CLC and handle execution of VMs on NCs
Storage Controller (SC)	Provides persistent block storage to the instances and allow creation of snapshots on Walrus
Node Controller (NC)	Manages instance life cycle and send resource availability and utilization information to CC
VMWare Broker	Mediates interaction between CC and VMware hypervisor [Optional component]

3.2. Sniffer and Feature Extraction

Virtualization on the NC requires a bridged interface namely *br0*to be setup. All the local interfaces of the VMs then attach to the bridged interface in order to communicate with the external network. Ideally, for any packets coming through the bridged interface from the external network, it is forwarded to the iptables firewall within the Linux operating system. The security feature in Eucalyptus which allows VMs to be assigned a *security group* requires the security group rules to be implemented as firewall rules in the NC's local firewall – iptables.

Table 2. Captured Packet Fields

Destination IP Address
Source IP Address
Destination Port Number
Bytes of Data Transferred
SYN Bit Value
ACK Bit Value

In this proposed approach, a packet sniffer is placed on the NC's base operating system, which captures certain packet information for a certain time and then directs the sniffed packet fields to an extractor script to calculate and extract *time-based traffic flow features* for each IP address that is communicated through the bridged interface. This includes all packets flowing from the external network (outside Eucalyptus) to the NC and VMs, packets flowing from the NCs and VMs to the external network and packets flowing from VMs to VMs in Eucalyptus cloud. The sniffer used for the purpose of this research is a custom built sniffer, developed in C language using the *libpcap* library for packet capturing on Linux systems. For cloud providers, protecting tenants' privacy is paramount therefore a balance was reached between the volume and nature of data captured to protect the tenants from DoS attack while at the same time safeguard the

privacy. After careful consideration the data capture had been limited to 6 TCP fields (Table 2), completely leaving the TCP payload untouched.

The packet fields captured by the packet sniffer is input to the feature extraction script which calculates the traffic flow features (Table 3). These features are then used by the classifier to differentiate between legitimate and malicious traffic.

Table 3. Traffic Features in Traffic Flow

#	Feature Shortname	Feature Description
1	Bytes_In	Bytes incoming per unique IP
2	Avg_Bytes_In	Average bytes incoming per unique IP
3	Count	Number of occurrence for an incoming IP
4	Avg_Count	Average number of incoming IPs
5	All_Count	Total number of incoming IPs
6	Bytes_Out	Number of bytes sent to the incoming IPs
7	Avg_Bytes_Out	Average bytes sent to incoming IPs
8	Syn_bit	Sum of Synchronize bit values
9	Ack_bit	Sum of Acknowledgement bit values

3.3. Classifier

3.3.1. k-nearest neighbor

The k-nearest neighbor (kNN) is a simple, instance-based learning algorithm used for pattern recognition problems. kNN works by measuring the distance between a query instance and a set of instances in a dataset. The distance between two instances with *N* features such that x = {x_1, x_2, x_3..., x_N} and y = {y_1, y_2, y_3..., y_N} using Euclidean measure is:

$$d_E(x, y) = \sum_{i=1}^{N} \sqrt{x_i^2 - y_i^2} \quad (1)$$

3.3.2. Artificial neural network

An artificial neural network (ANN) is a set of processing units called 'neurons' when assembled in a closely interconnected network offers some features of the biological neural network [20]. ANNs have been adopted as classifiers for complex problems such as cancer classification [21] and image classification [22]. A feedforward ANN classifier has been used where the neurons are connected by links of weights with real values. Each neuron consists of a sigmoid activation function which transforms the weighted sum of the inputs producing an output from the neuron. The number of hidden layer neurons is determined as an average of the input and output layer neurons. *Back propagation* is used as the learning algorithm with *learning rate* set to 0.3 and *momentum* as 0.2. Fig.2 illustrates the ANN model used for experimentation.

An alert component is also integrated and is responsible for relaying DoS attacks detected by the classifier to the administrator for further action. Altogether, the sniffer, feature extraction process, classifier (kNN or ANN) and alert component depicts a real-time intrusion detection system.

4. Experimentation

This section covers the details of Eucalyptus cloud setup, the tools needed to simulate active users in the cloud, performing attacks and generating the dataset.

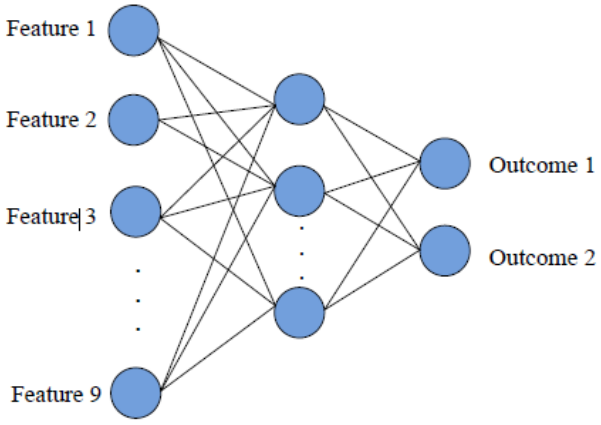

Fig. 2. Artificial neural network model.

4.1. Eucalyptus Cloud Setup

The cloud is deployed with Eucalyptus version 4.1 software. It runs on 10 physical servers and has a dedicated Storage Area Network (SAN) resulting in a computation capacity of 120 vCPUs and storage capacity of 7.3 TB. Each of the servers has the minimal version of CentOS 6.6 installed and the cloud components are installed on top of the operating system. The deployment of each Eucalyptus component is onto individual physical servers which provides these components dedicated server resources to carry out the functions. The SAN is used as a form of block storage for the SC. The cloud uses the KVM hypervisor to virtualize the computing, network and storage resources and deliver on-demand IaaS. Details of installing Eucalyptus cloud can be found in the Eucalyptus official installation guide [19].

Table 4. Security Groups for VMs

Security Group for Users and Attack VMs
Rule 1: Allow SSH (TCP Port 22)
Security Group for Webserver
Rule 1: Allow SSH (TCP Port 22)
Rule 2: Allow HTTP (TCP Port 80)

Eucalyptus cloud has to be configured to allow components to communicate with each other. The VMs once launched should be able to communicate with each other and the external network with a certain level of security in place. The *Edge networking* mode which is used in this setup removes the need to place a single Linux server in the data path for all VMs running in a single cluster and additionally removes the need to configure the underlying network to allow passing of VLAN tagged packets in the cloud [19]. A pool of private and public IP addresses is also required to be allocated to Eucalyptus.

Eucalyptus implements a feature known security groups in the cloud. Security groups act as virtual firewalls supporting ingress packet filtering for VMs they are applied to. Security groups are implemented from the CLC onto the VMs and is applied as iptables rules on the NCs. For the VMs on this test bed, two security groups are setup: one for the user and attack VMs and the other for the web server VMs (Table 4).

4.2. Testbed

In the testbed, different roles are assigned to VMs. As the research is based on VM to VM communication and attacks, it is necessary to simulate an active cloud environment where there are necessary VM to VM

communication. In the cloud, 3 types of VMs are setup - firstly 10 *m1.small* (256MB memory, 1 vCPU, 5 GB disk) instances of CentOS 6.5 and Ubuntu 12.04 are evenly setup as user VMs whose tasks is to continuously but randomly request for webpages from the webserver VM. The webserver VM is a *m3.xlarge* (2GB memory, 4 vCPU, 15 GB disk) instance which runs an Apache 2.2.15 webserver hosting a website of 6 webpages. 2 *m1.xlarge* (1GB memory, 2 vCPU, 10 GB disk) instances are setup as attack VMs whose tasks are to perform TCP SYN flood attacks based on defined times and scenarios.

Siege which is a webserver load test and benchmarking tool was used to simulate active and legitimate users of the webserver [23]. The Siege parameters were set such that for user VM 1 to VM 10, Siege would request for any of the 6 webpages from the webserver at a random time within a specific time. For example, for user VM 4, Siege would request for a webpage from the webserver any time between 20 seconds and then request for another webpage in the next 20 seconds.

4.3. TCP SYN Flood

TCP SYN flood attack uses the 3-way handshake connection establishment mechanism of TCP protocol to carry out the attack. In a normal scenario (Fig. 3a) the client would send a SYN packet to the server as a request to establish a connection with the server, after which the server would acknowledge the receipt of request and send a packet to the client with SYN and ACK bits set. At this point the server would have allocated resources for this client. The client finishes the handshake by sending the ACK back to the server. In a TCP SYN flood (Fig. 3b), the client, either with its own IP address or spoofed address(es) send multiple SYN packets to the server at a very high rate. Once the server receives these SYN packets, it allocates resources on the server and sends a packet with SYN and ACK bits set to either the client or to the spoofed addresses. The client or the spoofed address(es) however do not respond to the server with the ACK packet.

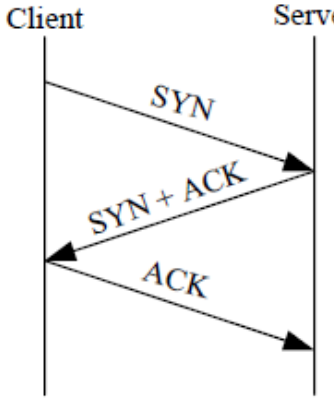

Fig. 3a.Normal TCP 3-Way Handshake

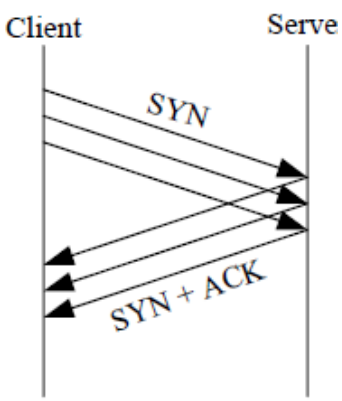

Fig. 3b.TCP SYN flood (No ACK from Client)

To carry out TCP SYN flood attacks, the hping3 [24] tool is utilized which allows customized SYN packets to be send to the server. Four scenarios of the attacks are carried out over a total duration of 4 hours and the details are summarized in Table 5.

Table 5. Attack Scenarios

#	Attack VMs	Data in Packet	Attack duration
1	1	40 bytes/packet	6 x 5 min attack for 1 hour
2	1	1500 bytes/packet	6 x 5 min attack for 1 hour
3	2	40 bytes/packet/VM	6 x 5 min attack for 1 hour
4	2	1500 bytes/packet/VM	6 x 5 min attack for 1 hour

5. Dataset

In the 4 hour experimentation which involved user VMs sending legitimate requests and attack VMs sending malicious requests to the webserver VM, the resulting outcome of the sniffer and the feature extraction process yielded a dataset of 5026 instances x 9 features. From this 5026 instances, 4391 instances were from user VMs while 635 instances were from attack VMs. The difference in the two classes of legitimate and malicious traffic from VMs is due to the fact that there were 10 VMs sending legitimate traffic while only 2 VMs were sending malicious traffic and the sniffer and the feature extraction process function based on capturing and extracting features for each incoming IP address in the traffic flow for the specified time internal of 5 seconds.

The original dataset is split into a ratio of 4:1 representing the training and test dataset groups. After splitting, the training dataset consists of 4021 instances of which 3513 instances belong to the legitimate class while 508 instances belong to malicious class. The 1005 instances belonging to the test dataset consists of 878 and 127 instances from the legitimate and malicious classes respectively. An extract of the original dataset is provided in Table 6. It contains 10 instances x 9 features and an additional column which labels the instances as either malicious or legitimate.

Table 6. Dataset Generated (Extract)

Bytes_In	Avg_Bytes_In	Count	Avg_Count	All_Count	Bytes_Out	Avg_Bytes_Out	SYN_bit	ACK_bit	Outcome
1457	2005.8	5	6.6	33	437	542.4	1	5	Legitimate
3991	2409	11	7.33333	22	874	578.333	2	11	Legitimate
2047	952856	6	21649.9	194849	373	4.75E+07	1	6	Legitimate
1612	2154.5	3	5.25	21	332	443.5	1	3	Legitimate
7923	4259.33	23	13	39	1747	1074	4	23	Legitimate
7446460	1.72E+06	169244	39091.2	195456	14620320	3212179	169175	0	Malicious
2684	1430.25	61	18.75	75	0	218.5	61	0	Malicious
264	2402.6	6	8.2	41	0	558.4	6	0	Malicious
3453600	1921821	78497	43676.8	218384	8674440	3.72E+06	78430	0	Malicious
4540136	2330457	103185	52956.4	264782	9278560	3.49E+06	103184	0	Malicious

6. Results and Discussion

6.1. Benchmarking Webserver

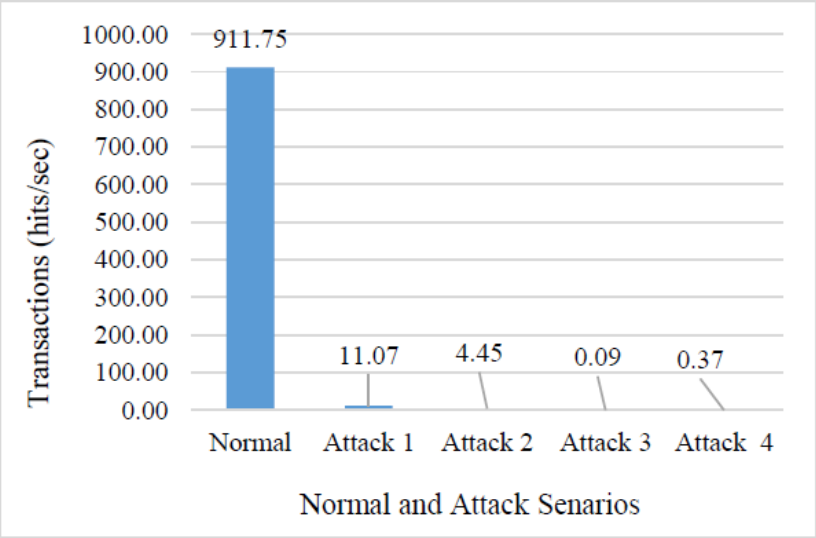

Fig. 4. Performance degradation of webserver.

As the aim of any DoS attack is to exhaust a system's resources such that it unable to provide service to legitimate clients, the TCP SYN flood attack aims for the same. The performance of the webserver VM which was under attack by one and two VMs (simultaneously) can be known by measuring transactions from an arbitrary client requesting for service when the webserver is under attack and when it is not. Such a task can be performed by Siege which has a benchmarking feature where the VM running Siege sends requests to the webserver without any delay between the requests. For a normal scenario where there are no attacks on the webserver but only legitimate requests by the user VMs and the four scenarios where the webserver is under attack, the benchmark results are shown in Fig. 4.

Fig. 4 shows the effect of TCP SYN flood attacks on the webserver in responding to legitimate requests by the benchmarking VM. In a normal scenario where there are only legitimate requests coming from user VMs, it is observed that on average 911.75 transactions or hits can be carried out by the benchmarking VM in a second. Performance of the webserver however deteriorates significantly with all attack scenarios. The performance of the webserver is affected most for attack scenario 3 where the webserver is slower by 10,131 times compared to the benchmarked normal rate. All transaction or hit values for all scenarios are based on averages obtained from five tests in each scenario.

6.2. Webserver System Resources and Bandwidth

This section looks at the effects that occur on the webserver under the TCP SYN flood attacks with respect to system load, memory usage and bandwidth utilization.

The webserver VM was setup with a memory size of 2 GB and processing capacity of 4 vCPU. When a TCP SYN flood of a high magnitude of such was carried out, it was presumed that the server would be exhausted of memory and processing with this attack. However, empirical observations show that there is no significant change on these two system resources when handling attack traffic. Fig.5a and Fig.5b are obtained by monitoring the memory and system load on the webserver. Fig.5a shows that there is slight increase in memory usage when the 5 minute TCP SYN flood attacks are in progress, however this is not a significant change from the normal memory usage. Fig.5b shows that the system load is not significantly affected by the attacks. The observations presented are for first 30 minutes of attack scenario 4. Similar results are observed for other attacks.

A significant change in bandwidth utilization is observed in all four scenarios as the attack VMs flood the webserver with SYN packets using 40 –1500 bytes of data in the packets. From Table 7, it is clear that one of the symptoms of anon-going TCP SYN flood attack is much higher incoming and outgoing network traffic compared to normal, however this should be differentiated from cases of *flash events* in which a large number of legitimate clients simultaneously access a webserver [25], which could create some false positive identification by an intrusion detection system.

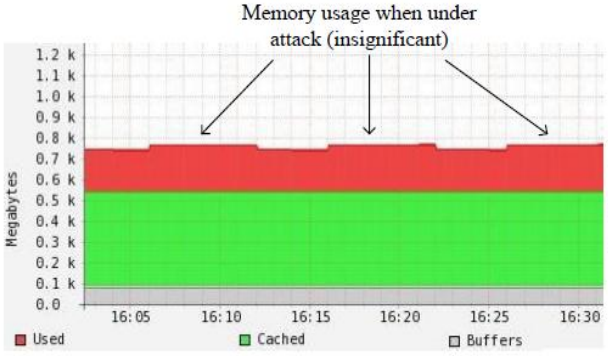

Fig. 5a. Webserver memory usage.

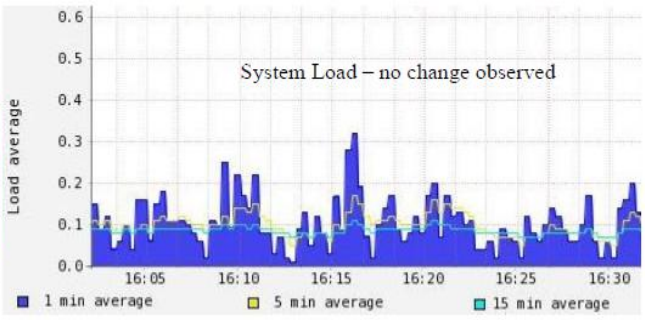

Fig. 5b.Webserver system load.

Table 7. Webserver Bandwidth

	Normal	Attack 1	Attack 2	Attack 3	Attack 4
Incoming (max)	1 kB/s	5222 kB/s	101626 kB/s	5844 kB/s	143588 kB/s
Outgoing (max)	3 kB/s	4272 kB/s	3874 kB/s	6070 kB/s	5459 kB/s

6.3. Classification Accuracy

The kNN and ANN classifiers are first subjected to a training phase with 5-fold cross validation which is mainly performed to prevent the classifiers from overfitting. The goal of the training phrase is to identify the best classifier model of kNN and ANN such that the instances from the test dataset can be subjected to this model. For the kNN classifier the training dataset is evaluated using k = {1, 3, 5, 7, 9} neighbors thus forming 5models of the kNN classifier. Two different ANN classifier models are used by having 1 and 2 hidden layers in the two ANNs. For ANN model 1 where 1 hidden layer is used, 5 neurons are placed on this layer for computation. For ANN model 2, the 2 hidden layers contain 7 and 4 neurons respectively. Increasing the number of hidden layers can enable the classifier to get better accuracy at an increased cost of computation and possible overfitting.

In this classification problem, the effectiveness of the classifier model is determined by a single *F1 Score*. The F1 score strikes a balance between the *precision* and *recall* values for a classifier. The precision of a classifier model is defined by the number of true positives divided by the number of true positives and false positives and recall is defined as the number of true positives divided by the number of true positives and the number of false negatives. An instance is considered to be true positive classified if it is malicious and classified as malicious, true negative classified if it is legitimate and is classified as legitimate, false positive classified if it is legitimate and is classified as malicious and false negative if it is malicious and is classified as legitimate by the classifier. The F1 score is thus calculated as follows:

$$F1\ Score = 2\ \frac{precision\ x\ recall}{precision + recall} \tag{2}$$

The precision, recall and F1 scores have been calculated for the 5 kNN and 2 ANN classifier models, and the results are presented in Table 8and Table 9respectively. By comparing the F1 scores it is found that the ANN models have the highest F1 scores and it is the same for 1 and 2 hidden layer models. As the 1 hidden layer model takes less computation time it is chosen over the 2 hidden layer model to be used for running test cases.

The precision, recall and F1 scores have been calculated for the 5 kNN and 2 ANN classifier models, and the results are presented in Table 8and Table 9respectively. By comparing the F1 scores it is found that the ANN models have the highest F1 scores and it is the same for 1 and 2 hidden layer models. As the 1 hidden layer model takes less computation time it is chosen over the 2 hidden layer model to be used for running test cases.

Table 8. Precision, Recall and F1 Scores for kNN Classifier Models (Training)

	Precision	Recall	F1 Score
k=1	0.969	0.97	0.97
k=3	0.947	0.955	0.951
k=5	0.957	0.961	0.959
k=7	0.945	0.951	0.948
k=9	0.938	0.951	0.944

Table 9. Precision, Recall and F1 Scores for ANN Classifier Models (Training)

	Precision	Recall	F1 Score
h=1	0.998	1	0.999
h=2	0.998	1	0.999

When the 1 hidden layer ANN model is subjected to a test dataset of 1005 instances of which 878 instances are from the legitimate class and 127 are from the malicious class, the classifier gives a F1 score of 1. This is the maximum desired F1 score for the test dataset and implies a 100% classification accuracy for all test instances. Table 10 shows the result obtained during the test run. As the test instances were not part of the training phase, and thus were unknown to the classifier, it implies that the classifier had learnt to classify malicious traffic instances from legitimate ones during the training phase and then appropriately classified the given test instances.

Table 10. Precision, Recall and F1 Scores for ANN Classifier (Test)

	Precision	Recall	F1 Score
h=1	1	1	1

The 100% classification by the ANN classifier is credited to the design of the DoS detection mechanism and the quality of features extracted from the traffic flow. The design of the mechanism permits the traffic flow to be discretized into traffic flow instances with distinct features describing the flow. Out of the 9 features in each flow, the SYN_bit feature is of most significant for the classifier in learning the behavior of legitimate and malicious traffic flows. When an attacker executes a TCP SYN flood, a high volume of packets with the SYN bit set is received on the client machine. Thus by extracting the SYN bit field from the packets we increase the classifiers' ability for detection of such attacks.

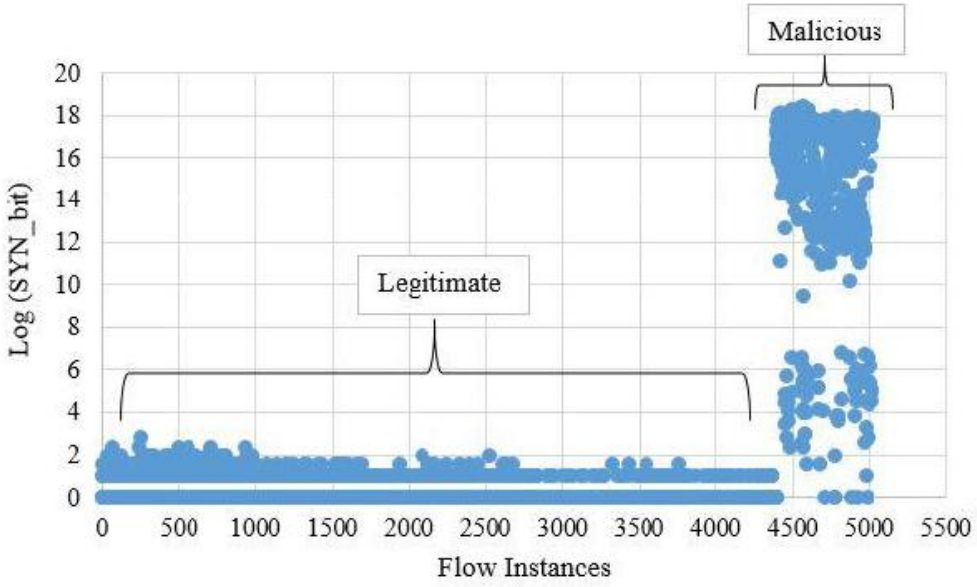

Fig. 6. SYN bit count in flow instances.

Fig. 6 illustrates the high volume of SYN bit count for the malicious traffic flow instances compared to a low count for legitimate traffic flow instances for the dataset used.

7. Conclusion

The research work carried out addresses a critical issue of VM to VM DoS attacks in a cloud environment. An IaaS cloud environment, itself represents a virtual network and thus cloud providers need to provide significant level of security to its clients in this network. Relying on security groups of Eucalyptus or AWS alone is not sufficient to protect client VMs from DoS attacks such as the TCP SYN flood attack. With strict security group rules and SYN cookies enabled on the webserver VM, it was found that TCP SYN flood attacks can still deteriorate the performance of a webserver VM running Apache, making it up to 10,100 times slower than its normal capacity to respond to requests.

An approach involving a packets sniffer, feature extraction process and a machine learning algorithm has been proposed. Upon training machine learning algorithms with features of legitimate and malicious traffic flow instances, the machine learning algorithm can identify malicious traffic from legitimate ones. The kNN an ANN classifier models had been evaluated during the training phase and the ANN classifier with 1 hidden layer was chosen for the test phase based on the F1 score. The chosen ANN classifier model was able to provide a F1 score of 1 with the unknown test instances implying a 100% classification accuracy for all test instances that was obtained from the dataset generated from the testbed.

The proposed framework has showed positive results with TCP SYN flood attacks and additionally the framework can be experimented with other types of DoS attacks like the ICMP flood and Slowloris attack, such that with a security mechanism multiple types of DoS attacks can be detected in the cloud environment. In addition, future work on this research can focus on *incremental learning*. As traffic flow changes over time based on certain factors such as increased number of users to a website, an updating mechanism can be run on certain time intervals allowing the classifier to learn any new traffic flow patterns. Dua and Du [26] also present similar arguments, stating that cyber-infrastructures experience huge amounts of continuous and dynamic data and thus incremental learning is essential for machine learning algorithms.

References

[1] Gupta, S., Kumar, P., & Abraham, A. (2013). A profile based network intrusion detection and prevention system for securing cloud environment. *International Journal of Distributed Sensor Networks*.

[2] Zhang, Q., Cheng, L., & Boutaba, R. (2010). Cloud computing: State-of-the-art and research challenges. *Journal of Internet Services and Applications, 1*, 7-18.

[3] Amazon EC2. Amazon Web Services. (2016). Retrieved, from the website: https://aws.amazon.com

[4] Microsoft Azure. Microsoft. (2016). Retrieved, from the website: https://azure.microsoft.com

[5] Salesforce. (2016). Retrieved, from the website: https://www.salesforce.com

[6] HCL Technologies. (2014). Whitepaper — Cloud services for energy and utilities. *HCL Technologies*, Noida, Uttar Pradesh, India.

[7] Cisco Systems Inc. (2014). *Cisco 2014 Annual Security Report*. Cisco Systems Inc. Americas Headquarters, San Jose, CA, USA.

[8] Juniper Network Inc. (2013). White paper — Defending against application-layer DDOS attacks. *Juniper Network*, Inc., Sunnyvale, CA, USA.

[9] DDoS Attacks, Incapsula. (2016). Retrieved, from the website: http://www.incapsula.com/ddos/ddos-attacks/

[10] Shea, R., & Liu, J. (2013). Performance of virtual machines under networked denial of service attacks: Experiments and analysis. *IEEE Systems Journal*, *7*, 335-345.

[11] Lonea, A. M., Popescu, D. E., Prostean, O., & Tianfield, H. (2013). Evalution of experiments on detecting distributed denial of service (DDoS) attacks in eucalyptus private cloud. *Soft Computing Applications,*

Advances in Intelligent Systems and Computing, 195, 367-379.
[12] Snort. (2016). Retrieved, from the website: https://www.snort.org/
[13] OpenNebula. (2016). Retrieved, from the website: http://opennebula.org/
[14] Modi, C., Patel, D., Borisaniya, B., Patel, H., Patel, A., & Rajarajan, M. (2012). A survey of intrusion detection techniques in Cloud. *Journal of Network and Computer Applications, 42*, 42-57.
[15] Engen, V., Vincent, J., & Phalp, K. (2011). Exploring discrepancies in findings obtained with the KDD cup 99 data set. *International Journal of Intelligent Data Analysis, 15*, 251-276.
[16] Singh, S., & Bansal, M. (2013). Improvement of intrusion detection system in data mining using neural network. *International Journal of Advanced Research in Computer Science and Software Engineering*, *3*, 1124-1130.
[17] IETF RFC 4987: TCP SYN flooding attacks and common mitigations, internet engineering task force (IETF). (2016). Retrieved, from the website: https://tools.ietf.org/html/rfc4987
[18] Hewlett-Packard. HP Helion eucalyptus. (2016). Retrieved, from the website: http://www8.hp.com/us/en/cloud/helion-eucalyptus-overview.html
[19] Eucalyptus documentation. (2016). Retrieved, from the website: http://docs.hpcloud.com/eucalyptus/4.1.2/
[20] Rani, K. (2011). Analysis of heart diseases dataset using neural network approach. *International Journal of Data Mining and Knowledge Management Process*, *1*.
[21] Kumar, R., Chand, K., & Lal, S. P. (2014). Gene reduction for cancer classification using cascaded neural network with gene masking. *Advances in Artificial Intelligence*.
[22] Rashmi, S., & Mandar, S. (2011). Textural feature based image classification using artificial neural network. *Advances in Computing, Communication and Control.*
[23] Wang, P., Huang, W., & Varela, C. (2010). Impact of virtual machine granularity on cloud computing workloads performance. *Proceedings of the 11th IEEE/ACM International Conference on Grid Computing,* Brussels
[24] Sanfilippo, S. H. (2016). Retrieved, from the website: http://www.hping.org/
[25] Sachdeva, M., & Kumar, K. (2014). A traffic cluster entropy based approach to distinguish DDoS attacks from flash event using DETER Testbed. *ISRN Communications and Networking*.
[26] Dua, S., & Du, X. (2011). Classical machine-learning paradigms for data mining. *Data Mining and Machine Learning in Cybersecurity*.

Raneel Kumar received his BScGCEd degree in 2012 from the University of the South Pacific (USP), Fiji and received his MSc degree in Computing Science in 2017 at the same university. He works as a User Consultant for the School of Computing, Information and Mathematical Sciences at USP. His research interests are in the field of cloud computing, cybersecurity and machine learning.

Sunil Lal received his BSc degree in 2001, and MSc degree with gold medal award in 2005, from the University of the South Pacific (USP), Fiji. In 2009, Sunil Lal received his Ph.D and a president's honorary award for scholarly achievement and contributions to the university community from the University of the Ryukyus, Japan. He is currently a Lecturer with Massey University, New Zealand. His research interest is in the field of machine learning, and robotics, and he has served as an editorial board member and reviewer for various journal and conferences.

Alok Sharma received the B.Tech. degree from the University of the South Pacific (USP), Suva, Fiji, in 2000 and the M.Eng. degree, with an academic excellence award, and the Ph.D. degree in the area of pattern recognition from Griffith University, Brisbane, Australia, in 2001 and 2006, respectively. He was with the University of Tokyo, Japan (2010–2012), as a Research Fellow. He is an Assoc. Prof. at the USP and an Adjunct Assoc. Prof. at the Institute for Integrated and Intelligent Systems (IIIS), Griffith University. He is also a Scientist at RIKEN Center for Integrative Medical Sciences, Japan. He participated in various projects carried out in conjunction with Motorola (Sydney), Auslog Pty., Ltd. (Brisbane), CRC Micro Technology (Brisbane), the French Embassy (Suva), and JSPS (Japan). His research interests include pattern recognition, computer security, human cancer classification, and protein fold and structural class prediction problems. He reviewed several articles and is in the editorial board of several journals.