Intrusion Detection by Forensic Method in Private Cloud using Eucalyptus

Mayur S. Patil
Dept. of Computer Engineering,
MAEER's MIT College of Engineering,
Kothrud, Pune.
India

Prof. Bharati Ainapure
Dept. of Computer Engineering,
MAEER's MIT College of Engineering,
Kothrud, Pune.
India

ABSTRACT
Cloud computing has become the mature term which has dealt from single user to large enterprises. The private cloud platform building framework Eucalyptus has great pace of development within short span of time. Achieving AWS (Amazon Web Services) compatible features development along with scalability and sustainability has introduced several issues have an adverse effect on the cloud system. In continuing with this, the chances of intrusion also increase evading traditional mechanism of security. Issues have been introduced due to seamless integration of such structure with computing technologies and so on. By taking advantage of such flaws, the Cybercrime is rapidly increasing in this field. The proposed work is regarded with Digital forensics technique and intrusion detection mechanism. In this scope of work, an experimental setup of Eucalyptus with Snort NIDS (Network Intrusion Detection System) to detect attacks using snort rules has been created. The Eucalyptus Cloud components and Snort logs are exported to outside cloud network to rSyslog server which would be later analyzed by the Awstats log analyzer. Accompanied to above, this scope of work also addresses toward the issue of Eucalyptus to export its logs to the remote rSyslog server. This system will definitely help to reduce the strain on the Cloud forensics.

General Terms
Private Cloud, Intrusion Detection System, Log Analysis, Digital Forensics.

Keywords
Eucalyptus, Cloud Forensics, IDS, Snort, Logs, Awstats log analyzer.

1. INTRODUCTION
The Cloud Computing has spread like conflagration in the market within short duration travelling from buzzword to groundbreaker. Now, there are even small companies from animation to distributed computing, though gradually; have started their move to conform changes favored for Cloud field. Looking at the alternative side of the issue, the most concerned part of any system when brought into production environment is security. Security is a proactive approach to protect the existing system from intruders and cloud forensics. Following graph indicates Google trend analysis which typifies the worldwide statistics of the Cloud Forensics from April 2009 to November 2013. The hike is actually observed in the middle of year 2010-11 indicated by the curve of the red line.

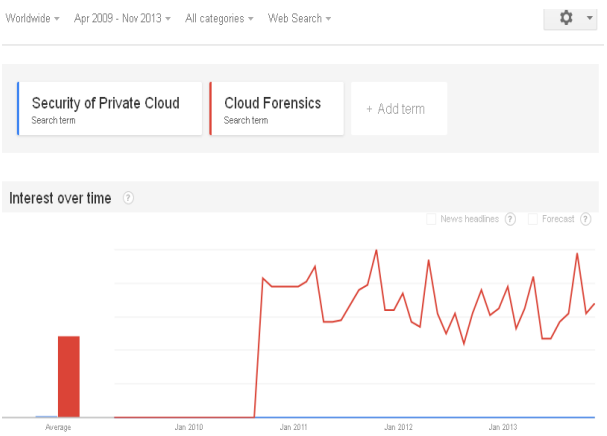

Fig. 1 Cloud Forensics Statistics

There are various open source private cloud platforms like Eucalyptus, OpenNebula, Openstack, Nimbus etc. Among the crowd, Eucalyptus, is the AWS (Amazon Web Services) compatible private and hybrid cloud building framework for this scope of work. Setting Eucalyptus means to use Amazon Web Services (AWS) within the private cloud environment. It has observed since from 2008, Eucalyptus is having the fast pace of development proven from its features like Elastic Load Balancing, autoscaling, cloudwatch [1]. These types of features need flexible architecture to extend provision of services programmatically. Eucalyptus has built on various technologies mainly C, Java, JavaScript, Groovy and Python [2]. Issues to be underlined are technology amalgamation [3] deals with a mixture of two or more technologies in one environment, xml security issues, for example, in Eucalyptus under the CLC source code tree there is file xml-security-config.xml. There is a tag for TransformAlgorithms, if remain uncommented, could facilitate the DoS (Denial of Service) attack on Cloud system; inherent security issues [4] [25] etc. which has divulged security flaws in Eucalyptus to bring infliction for the security persons. Due to spry development in technology, attackers have found new ways to intrude to the high storage publics private information and known to be the great cloud service providers like Facebook, Twitter reported as "under attack" many times despite of proficient security geeks. As intruders/attackers are evading security criterion, it has observed that to finagle with these security issues the mainstream methodologies of security seem to be superannuated. Even the most of known attacks like DDoS (Distributed DoS) flood, SSL (Secure Socket Layer) DoS, Land Attack, TCP SYN flood alias hping3 flood etc. have the power to bring down the whole system due to belittle misconfiguration in software/architecture. These consequences laid security analyst to make afterthought on

established contrives of the security mechanism of cloud systems. For ex, if an attacker is able to find any programmatic or configuration based loophole in the system, it will easier for him to intrude and put whole system either down or get crashed.

In private cloud environment, it is almost Laputan to forbid all real time intrusions because of distributed, dynamic and virtualized nature of Cloud which has hindered the itinerary of investigation for forensics analysis. So, there is high hike seen in the field of Cloud forensics. The foundation of cloud forensics is mainly based on Digital forensics as shown below:

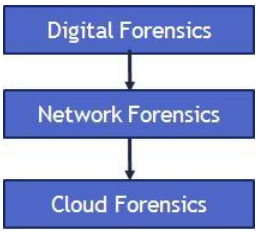

Fig 2: Hierarchy of Cloud Forensics

Cloud forensics is the application of digital forensics in the cloud computing field with a pure inheritance from the Network forensics [5]. So, here, deliberation is that the Digital forensics techniques will be applicable in case of Cloud forensics. The traditional approach which is either limited to mathematical models or having very high degree of implementation could be difficult to understand by entrant researchers. So, rather than reinventing the wheel, two well-known and reliable methodologies of digital forensics has used in this proposed work. Considering a scenario that if an attacker has gained the access of the system, he could easily compromise the authentic areas of the system. From the evidence point of view in Cloud forensics, logging is only exclusive technique to become evidence of activities performed in a distributed and dynamic environment like the cloud. To simplify logging, timeline analysis has used to get results in an apprehensible manner to get a quick analysis of logs.

The scope of the paper is described as follows: Section 2 deals with the related work, Section 3 deals with experimental setup & proposed scenario, Section 4 deals with experimental observations, Section 5 having Conclusion, Section 6 deals with future scope.

2. RELATED WORK

The aim behind this work is to find the vulnerabilities & introduce a forensics methodology which could be helpful to resolve issues affecting the Cloud forensics related to IaaS platform. The focus of this work is on intrusion detection with generation of attack dataset using real time traffic and IDS logs by exporting it to outside the victim network. Various approaches discussed for intrusion detection such as mathematical models like chi-square test, Hidden Markov Model based IDS [6]; frameworks discussed like, random forest tree based analysis [7]; architectures like NICE [8], multilevel intrusion detection [9]; methods like multi-variant correlation analysis [10], Poisson distribution [11]. Commencing with the survey, let's take a tour of Digital forensics & its technique's pros and cons first. The "Digital Forensics" term has a wide range of use as well as implementation. From expert perspective, it is proactive approach i.e. defensive approach to analyze and protect the system from being gets compromised in the future but most of the CSP (Cloud Service Provider)'s look at it from reactive approach i.e. to heal the system after being compromised. Increasing criminal activities have put gainsays in front of analyst because intruders/attackers always pinging for recent vulnerabilities in software; no matter how advanced they are [12]! In private IaaS based clouds, it has been observed the scenario that the on-site virtualized environment introduced risk with data mainly from the vendor itself. There are two possibilities, either CSP is involved in breaching user-based policies or he gets compromised. The term "on-site virtualized environment" means the CSP permits partaken of either same or multiple resources to user groups. While investigating for compromised user data, this policy introduces risk of exposure of other users' data to the analyst due to shared nature [3] [13] [26]. The more fruitful analysis is found to be Live analysis but on the other side most agile in nature. The investigator is "The Lucky One" if he has such opportunity because in a cloud environment, it is nearby impossible to work with affected data in the runtime. Introduction of Virtualization puts limits on traditional forensics. For example, if a user is using any application software, as soon as user exists from the virtual environment, his data gets lost and it is difficult to track back any activity/transaction that he has done earlier because many machines involved in this process. Though virtual introspection (VI) seems to be the solution but managing a second VM with compromised hypervisor seems to be challenging [14] [15] [16]. Other concerns are regarding the cross border red tape legislations which could have investigators, users and vendors hands' tied as per norms and conditions. The conflicts of two policies make this issue more ambiguous. For example, the data protection acts put limits on investigation difficult to access data in time. Globally standardized legal agreements are seem to be a solution; but not yet enforced [4] [15]. Evidence collection, identification, preservation, analysis, reconstruction and reporting are the basic steps of traditional forensics procedure. Techniques such as a VPN tunneling approach, Image recording of data also try to reveal strain on forensics system [17].

For forensics investigators, looking from insider's i.e. CSP's view, certain things need to be reconsidered. It includes CSP's operating modes of users' data inside cloud environment, which affects user's trust, resource sharing and attack strategies. These things could affect the profession as well as economic status of CSP but same issue obstructs here i.e. legal and jurisdictional considerations. The approach is discussed of tags of resource allocation and migration, but again seems to fail due to standardized environment [24]. If the data protection has to achieve in a Cloud, automatically how data travels to and from that position, here is system, needs to be considered. That's where Network forensics challenges come into the picture! It includes a single chance of network traffic analysis, touchstone routines which require to execute a complete set of forensic procedure but these steps seems too immature to criticize the concerns regarding the virtualization; the power intromittent to Cloud computing [17].

The analysis of attacks becomes easy when IDS has been implemented in Cloud environment. Consider the multilevel IDS and log system, the key points for efficient design are the group-based authentication and rule-based leveling with anomaly detections. But this model is having lacunae of managing the problem of resource consumption when the rule-based analysis is in full mode and due to common authentication policies, the chances of exposure to other people data in group to the forensics investigator if one of the

members from the group gets compromise [9]. On the other hand, document cluster approach in the digital forensics scenario has tested with six data mining algorithms on five real world data sets which could be optimized with algorithmic parameters tuning but having high computational cost [18].

Moving to the next essential part that is IDS based approaches. IDS are mainly classified into three categories: HIDS (Host based), DIDS (Distributed) & NIDS. NIDS advantage is that it has more real time intrusion detection capability compared to others and accumulates activities across the network infrastructure at one central log server/location. Inside network, most of authentic connections are SSH based which prone to face SSH based dictionary attacks. The performance optimization of anomaly detection is based on IDS parameters tuning as well as optimization methodology to HMM (Hidden Markov Model) based IDS [6]. Mainly focused attack in the network is the DoS. To use the multivariate correlation analysis for detection purpose leads to use only patterns of network traffic consisting of normalized data put limits on analysis due to lack of tests with real world data sets [10]. For the low cost of an IDS, the parallel string machine scheme is useful to reduce strain on memory in the context of parallel string matching engines [19]. The time factor to compromise the system depends on Poisson process which heavily depends on number of intrusions & system level security on the client as well as server side [11].

To ameliorate intrusion detection on both front end & back end of web applications on network infrastructure, DoubleGuard IDS is proposed i.e. monitoring webserver at front-end and databases at backend but this approach has hindered due to dependency on static websites [20]. For enhanced detection rates, Discontiguous system calls are used but the overhead of building datasets offline with confined trained data is there [21]. IDS based architectures are used to improve inherent security in the cloud such as NICE [8] has introduced its sub-models to detect & mitigate attack consequences. While on a network sensor embedded NIDS [22] provides load balancing of network traffic by carry-over it to DIDS & provide on demand service but only limited in basic web service specifications. A framework which is most noticeable and typically IDS based is Random forests network. It works with three types of misuse detection, anomaly and hybrid but limits due to supervised anomaly detection [7].

From the survey, the point has coined that there is a noticeable difference between problems and techniques to solve them. So, Log analysis is the proficient technique for use in this case. It is the most possible and viable approach which seems to be most approved practice amongst forensics analysts [4] [12] [17] [23]. Also, log analysis is one of the prioritized methods of forensic methods. It is useful for pattern recognition, normalization etc. It is immensely helpful about revealing information regarding past and present state of the system. Logging mechanism is proved helpful for managing and troubleshooting of IDS, Clustering techniques, Database transactions and so on.

During deciding strategies, there is brainstorming part that what approach should be followed for this; a framework or algorithm or architecture? These approaches have their own implementations in respective field. When they come in cloud arena, they throttle due to application level specifications. So, to speed up process blending of two random approaches that are logged and timeline analysis seems to be the most viable approach.

3. EXPERIMENTAL SETUP

3.1 Installation

The proposed work consists of test-bed of Snort with Eucalyptus Private Cloud. In this scope of work, results are obtained from experimental footsteps. They are based on observations as well as performance analysis which are tested and obtained from practical experimentations. All these experiments have performed with following setup specifications:

a. Under institute proxy network.

b. On standalone machines.

Setup consists of three machines from which two machines are having a Eucalyptus Private Cloud and third machine will act as duple which is outside the cloud network as:

- Public network client machine from attacking perspective.
- rSyslog server machine for log amassing and analysis.

The specifications are as follows:

Two CentOS 64 bits machines of Eucalyptus Private Cloud from which:

- First machine is of Eucalyptus Front end with Snort NIDS having 2 GB RAM, 300 GB HDD.
- Second machine with Eucalyptus Node controller (NC).

Third machine is Ubuntu, 32 bits, acts as rSyslog server with the Awstats log parser.

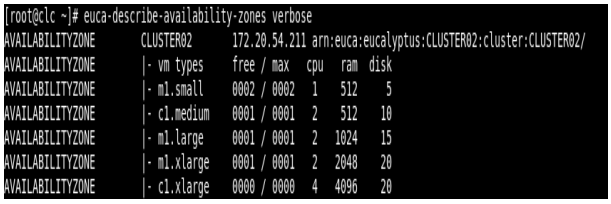

Fig 3: Eucalyptus Resources Status from Front end machine.

3.2 Implementations

The front end consists of the following components except NC:

- CLC (Cloud Controller)
- W (Walrus)
- SC (Storage Controller)
- CC (Cluster Controller)

CLC has selected as a target for attacks because CLC used to process requests of state management by the user. To make high performance for database access object persistence, it has become heavily multi-threaded and asynchronous.

- It is an entry gate in the Cloud.
- It is administered to perform any operation between end user and CC.

If one is able to bring down the CLC, the whole cloud system stops responding as well as the unavailability of resources from NC.
Here, two types of attacks has been executed on CLC which has been found to be simple but highly penetrable:

- TCP SYN flood attack

In this case, Transmission Control Protocol (TCP) based licit connection gets established from Clients and becomes difficult for the server to respond them if a flood of packets/postulations send uninterruptedly. Considering three modes of attacks [28], here, only two of them have used which are:

- Spoofing based Attack
- Distributed attack

These modes are able to make system unresponsive but did not get crash. The benefit of this attack is that the server is unable to allocate or provide resources to clients despite they are available in healthy condition.

- thc SSL DDoS Attack

In this part, the "thc SSL DoS" tool has initiated large SSL handshakes mechanism [29] with the server so that it will easy to bring down and eventually crash down the cloud system. The fundamental idea is to commit large numbers of handshakes with a server after establishing secure socket layer (SSL) connection. SSL is the protocol which is in the middle of Application and Network layer which ensures security and confidentiality.

Here, three phases of attack have performed as follows:

3.2.1. Phase-I: Exclusive TCP SYN Flood Attack:

The most popular & admin/hacker friendly utility hping3, basically used to test firewall rules and nmap protocol scanning purposes [27]. For example,

```
[root@clc] # hping3 -1 172.20.54.211
```

Before invoking TCP SYN flood, CPU statistics of front end, the system was receiving data at a rate of 71.4 KB/s as encircled in below image:

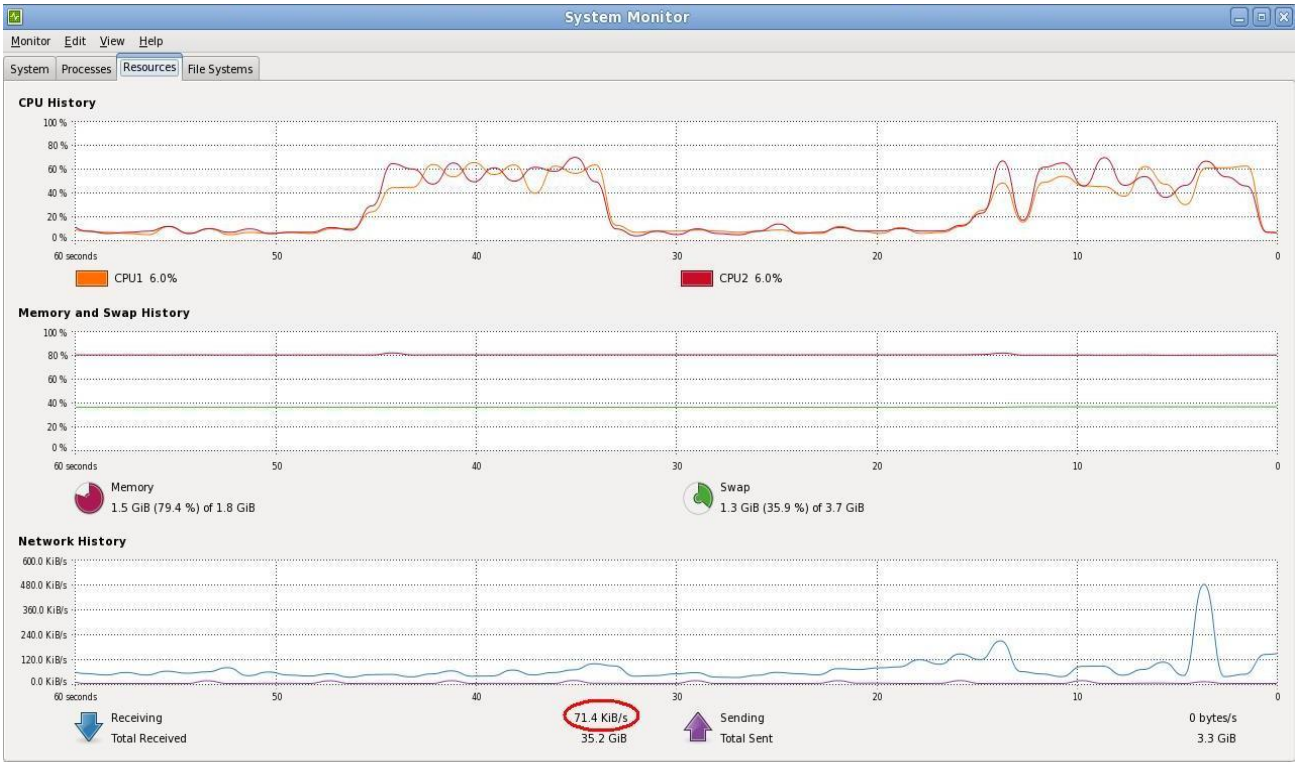

Fig 4: Legitimated Traffic (Before any attack) CPU Statistics

Exclusive testing for TCP SYN flood, three emplacements are used Logserver, NC machine and instance of image which is stored at the Walrus on Front end; as per following figure:

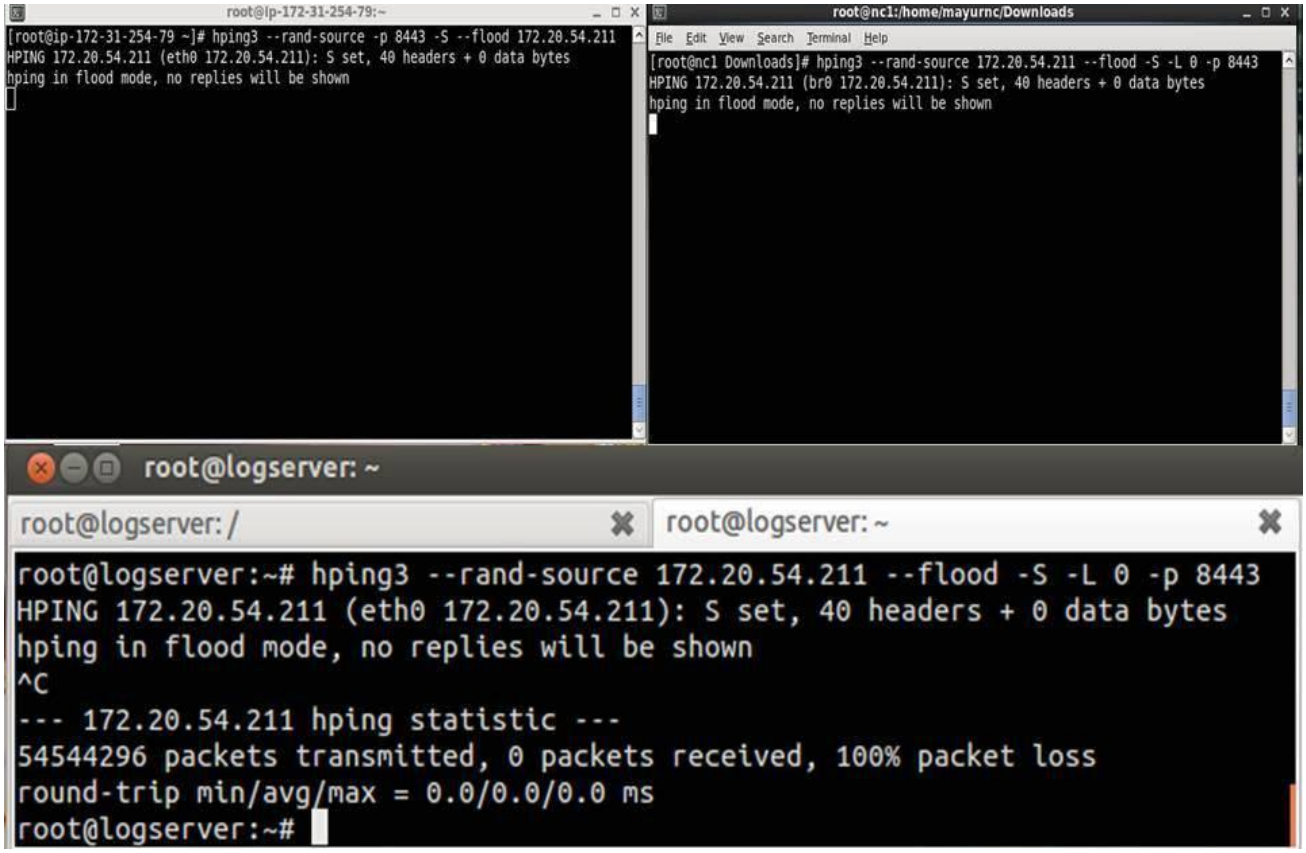

Fig 5: TCP SYN Flood from remote machine, NC and Logserver

In this case, spoofed as well as distributed attack strategy has observed. When this hping3 flood command based attack has instantiated, there has observed a sudden hike in the data receiving from the random IP address. The machine even stops responding even from opening the command line.

Fig 6: Resultant traffic after attack

The figure no. 6 is the clear indication of resulting incoming data receiving at the rate of 8.5 MB/s. In this case, the machine responds slowly but did not crash.

3.2.2 Phase-II: Distributed Attack (Internal & External Intrusion):

In this phase, SSL DoS as well as a TCP SYN flood has invoked to make cloud system unresponsive. This attack launched in multithreaded mode. Due to several SSL handshakes & flood of packets, it becomes easy to make a cloud system unprocurable when an attack has initiated even with basic security settings as shown in figure no. 8.

3.2.3 Phase-III: Distributed Attack (Internal Intrusion):

In this phase, the resources required for trespass has been optimized. So for that purpose, only NC has used here. As NC has a bridge with CLC, it uses a subnet of Eucalyptus CLC. After performing continuous attacks on CLC, it becomes unable to respond to attacks as shown in figure 9.

From these three cases, the conclusion can be drawn that if these two attacks are performed in multi-threaded manner, they are surplus to crash the entire cloud system less than 50 seconds despite of the firewall. As discussed before that it is unrealistic to analyze the distributed systems like a cloud by analyzing the real time traffic, it is viable to employ intrusion detection methodology rather than prevention. For this purpose, Snort is used. It is a NIDS which is useful to monitor and alert traffic in network if any suspicious behavior is observed and keeps logs of it for later analysis of the system. Among its configuration modes which are packet sniffing, packet logging, network intrusion detection (NID), Here NIDS mode has opted because this mode is handy to analyze traffic by cross checking it with customized shared object rules of Snort written in C language. Digging deep to understand the snort rules, there are two types of rules, text rules and shared object rules. Text rules are formed by parameters and their values. The sample text rule is as follows:

alert icmp $EXTERNAL_NET any → $HOME_NET any (msg: ”This is a sample rule for ICMP ping”; flow: established, to_ server; sid: 477; rev:1;)

This simple snort rule gives an idea about working of snort in detecting pinging client IP address and displaying the message. Then send message which is in double quote and flow of packets towards process using already logged vulnerabilities on snort website. The server direction with state of the connection specified. The attribute flow decides the direction of packets & state of connection with respect to server and client. The Snort id (sid) used to simplify the process using an id of already logged vulnerability on snort website. The attribute revision is the full form for rev.
The other type of rule is shared object (SO) rule. The steps of SO rules reverse that of text rules. These rules are written in C language using snort’s own data structure. After that, they have to compile with make tool which generates shared object based text rules. The difference is only that C language based SO rules are programmer friendly to add flexibility into rules while text rules are easy as they are also understood by non-programmers. It provides portability for users to use the rules. Though optimization has gained in case of the resources and abilities to bring down machine, three rules for DoS attack detection have been designed considering every possibility of an intrusion:

1. A TCP SYN DDoS flood of
   i. Random Ports
   ii. Internal Network
2. SSL DDoS from internal network.

The following diagrams are for the exported logs of snort alerts for intrusions:

```
6883 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     166.108.255.93:2397 -> 172.20.54.211:8443
6884 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     188.168.201.223:2398 -> 172.20.54.211:8443
6885 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     174.251.175.121:2399 -> 172.20.54.211:8443
6886 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     109.223.147.205:2400 -> 172.20.54.211:8443
6887 Nov 13 14:18:29 clc kernel: possible SYN flooding on port 8443. Sending cookies.
6888 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     183.62.2.191:2401 -> 172.20.54.211:8443
6889 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     116.151.222.87:2402 -> 172.20.54.211:8443
6890 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     109.142.170.147:2403 -> 172.20.54.211:8443
6891 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     223.93.243.158:2404 -> 172.20.54.211:8443
6892 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     0.232.164.33:2405 -> 172.20.54.211:8443
6893 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     183.62.2.191:2401 -> 172.20.54.211:8443
6894 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     116.151.222.87:2402 -> 172.20.54.211:8443
6895 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     109.142.170.147:2403 -> 172.20.54.211:8443
6896 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     223.93.243.158:2404 -> 172.20.54.211:8443
6897 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     0.232.164.33:2405 -> 172.20.54.211:8443
6898 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     174.0.202.95:2406 -> 172.20.54.211:8443
6899 Nov 13 14:18:29 clc snort: [3:100001:1] ###Possible TCP DDoS### [Classification: Attempted Denial of Service] [Priority: 3] {TCP}
     174.0.202.95:2406 -> 172.20.54.211:8443
```

Fig 7: TCP SYN flood from random ports

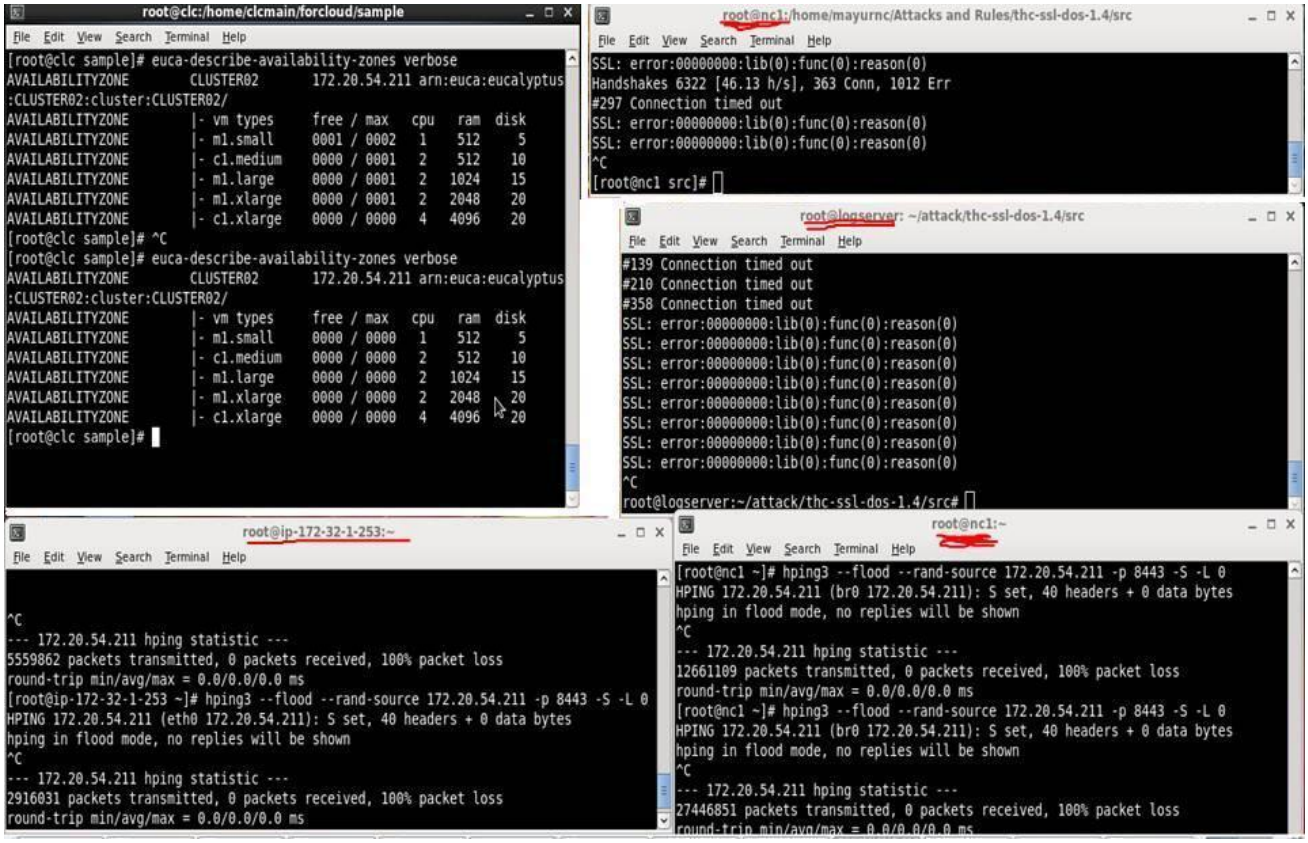

Fig 8: Phase-II: Distributed Attack (Internal & External Intrusion)

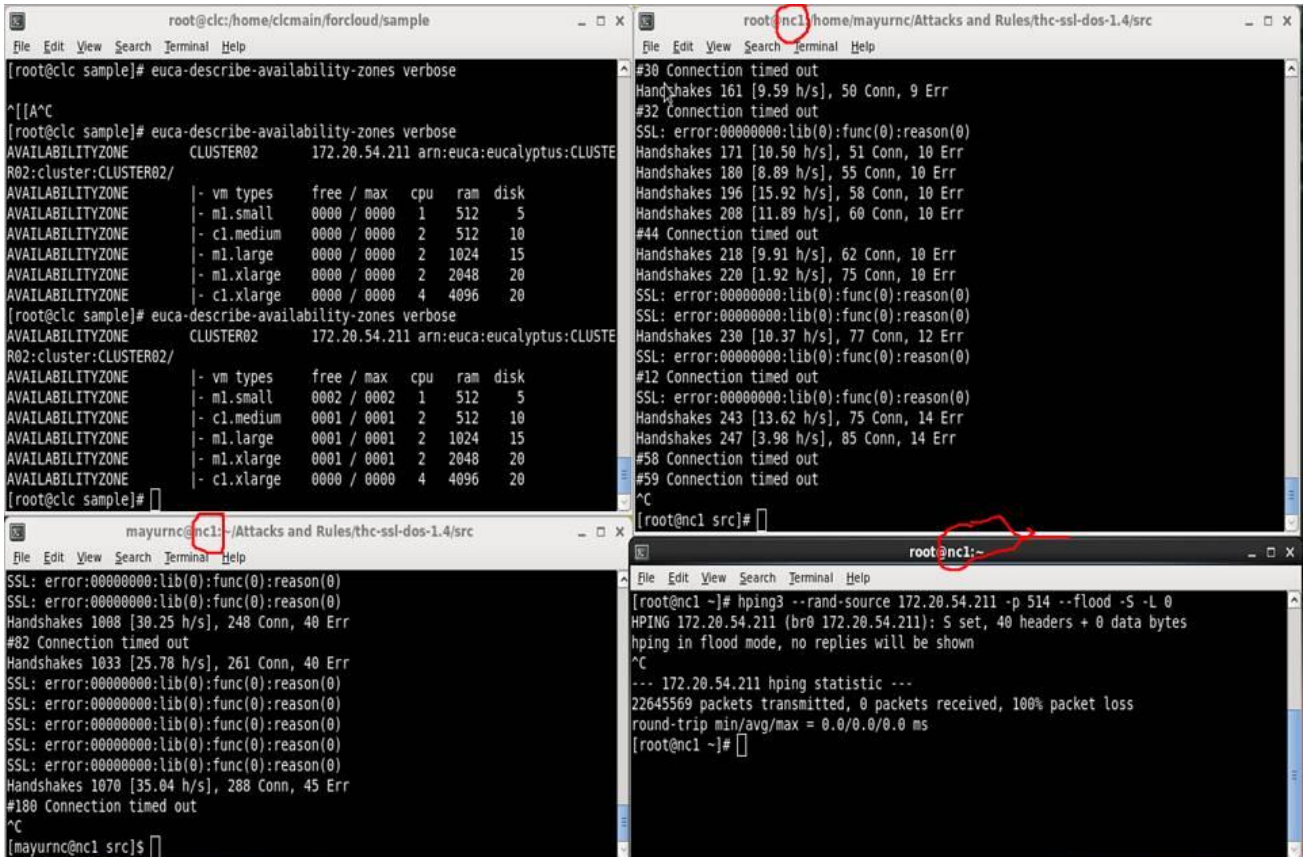

Fig 9: Phase-III: Distributed Attack (Internal Intrusion)

```
6301 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38419 -> 172.20.54.211:8443
6302 Nov 13 14:16:37 clc snort: [3:20437:1] DOS multiple TLSv1 Encrypted Handshake messages - THC-SSL tool, potential DoS [Classification:
     Attempted Denial of Service] [Priority: 2] {TCP} 172.20.54.212:38419 -> 172.20.54.211:8443
6303 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38419 -> 172.20.54.211:8443
6304 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38420 -> 172.20.54.211:8443
6305 Nov 13 14:16:37 clc snort: [3:20437:1] DOS multiple TLSv1 Encrypted Handshake messages - THC-SSL tool, potential DoS [Classification:
     Attempted Denial of Service] [Priority: 2] {TCP} 172.20.54.212:38420 -> 172.20.54.211:8443
6306 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38420 -> 172.20.54.211:8443
6307 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38421 -> 172.20.54.211:8443
6308 Nov 13 14:16:37 clc snort: [3:20437:1] DOS multiple TLSv1 Encrypted Handshake messages - THC-SSL tool, potential DoS [Classification:
     Attempted Denial of Service] [Priority: 2] {TCP} 172.20.54.212:38421 -> 172.20.54.211:8443
6309 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38421 -> 172.20.54.211:8443
6310 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38422 -> 172.20.54.211:8443
6311 Nov 13 14:16:37 clc snort: [3:20437:1] DOS multiple TLSv1 Encrypted Handshake messages - THC-SSL tool, potential DoS [Classification:
     Attempted Denial of Service] [Priority: 2] {TCP} 172.20.54.212:38422 -> 172.20.54.211:8443
6312 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38422 -> 172.20.54.211:8443
6313 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38423 -> 172.20.54.211:8443
6314 Nov 13 14:16:37 clc snort: [3:20437:1] DOS multiple TLSv1 Encrypted Handshake messages - THC-SSL tool, potential DoS [Classification:
     Attempted Denial of Service] [Priority: 2] {TCP} 172.20.54.212:38423 -> 172.20.54.211:8443
6315 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38423 -> 172.20.54.211:8443
6316 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
     172.20.54.212:38424 -> 172.20.54.211:8443
6317 Nov 13 14:16:37 clc snort: [3:20437:1] DOS multiple TLSv1 Encrypted Handshake messages - THC-SSL tool, potential DoS [Classification:
     Attempted Denial of Service] [Priority: 2] {TCP} 172.20.54.212:38424 -> 172.20.54.211:8443
6318 Nov 13 14:16:37 clc snort: [3:20438:1] On Euca Flood Detected [Classification: Attempted Denial of Service] [Priority: 2] {TCP}
```

Fig 10: Detection of TCP SYN flood & SSL DDoS attack from internal network.

In the above figure, two types of attacks are detected randomly because of packet analyzing method of snort rules. The SSL DDoS requires an SSL handshake mechanism which needs a much less number of packets compared to that of TCP SYN flood sends loads of packets which are quickly detected by snort due to the specification of packet rate in snort rules. Along with this, there is also work carried out on the Eucalyptus issue of syslog exportation as the part of community contribution. Eucalyptus is composed of mainly Java, C and Python on the basis of components. The following paradigms used by the eucalyptus component for log exportation:

- Java components (CLC, SC, W) use log4j module.
- C components (CC, NC) use eucalyptus. conf file and bash scripts.
- Python component (User console) uses the python logging module.

Following this methodology, exported logs of Eucalyptus cloud components have been obtained as shown below:

```
3992 Nov 13 14:06:31 nc1 euca-nc: physical memory available for instances: 3561MB
3993 Nov 13 14:06:31 nc1 euca-nc: virtual cpu cores available for instances: 2
3994 Nov 13 14:06:31 nc1 euca-nc: initializing backing store...
3995 Nov 13 14:06:31 nc1 euca-nc: disk space for instances: /var/lib/eucalyptus/instances/work
3996 Nov 13 14:06:31 nc1 euca-nc:                           044801MB limit (30.2140f the file system) - 860MB overhead = 43941MB = 42GB
3997 Nov 13 14:06:31 nc1 euca-nc:                           000000MB reserved for use (0.0140f limit)
3998 Nov 13 14:06:31 nc1 euca-nc:                           000000MB allocated for use (0.0140f limit, 0.015025070360f the file system)
3999 Nov 13 14:06:31 nc1 euca-nc:     disk space for cache: /var/lib/eucalyptus/instances/cache
4000 Nov 13 14:06:31 nc1 euca-nc:                           090962MB limit (61.3140f the file system)
4001 Nov 13 14:06:31 nc1 euca-nc:                           036882MB reserved for use (40.5140f limit)
4002 Nov 13 14:06:31 nc1 euca-nc:                           006588MB allocated for use (7.2140f limit, 4.415025070360f the file system)
4003 Nov 13 14:06:31 nc1 euca-nc: looking for existing domains
4004 Nov 13 14:06:31 nc1 euca-nc: no currently running domains to adopt
4005 Nov 13 14:06:32 nc1 euca-nc: /var/lib/eucalyptus/instances/cache: examined 12 blob(s) in 1 iteration(s): deleted 0, failed on 0 + 0, failed to open 0
4006 Nov 13 14:06:32 nc1 euca-nc:  VNET Configuration: eucahome=/,
4007 Nov 13 14:06:32 nc1 euca-nc:                      path=//var/run/eucalyptus/net,
4008 Nov 13 14:06:32 nc1 euca-nc:                      dhcpdaemon=,
4009 Nov 13 14:06:32 nc1 euca-nc:                      dhcpuser=,
4010 Nov 13 14:06:32 nc1 euca-nc:                      pubInterface=,
4011 Nov 13 14:06:32 nc1 euca-nc:                      privInterface=,
4012 Nov 13 14:06:32 nc1 euca-nc:                      bridgedev=br0,
4013 Nov 13 14:06:32 nc1 euca-nc:                      networkMode=MANAGED-NOVLAN
4014 Nov 13 14:06:32 nc1 euca-nc: spawning monitoring thread
4015 Nov 13 14:06:44 clc euca-cc: instances: 0000 (0000 extant + 0000 pending + 0000 terminated)
4016 Nov 13 14:06:44 clc euca-cc:     nodes: 0001 (0000 busy + 0001 idle + 0000 unresponsive)
4017 Nov 13 14:07:45 clc euca-cc: instances: 0000 (0000 extant + 0000 pending + 0000 terminated)
4018 Nov 13 14:07:45 clc euca-cc:     nodes: 0001 (0000 busy + 0001 idle + 0000 unresponsive)
4019 Nov 13 14:08:46 clc euca-cc: instances: 0000 (0000 extant + 0000 pending + 0000 terminated)
4020 Nov 13 14:08:46 clc euca-cc:     nodes: 0001 (0000 busy + 0001 idle + 0000 unresponsive)
4021 Nov 13 13:22:28 clc euca-console: INFO 200 POST / (172.20.54.213) 12.25ms
4022 Nov 13 13:22:28 clc euca-console: INFO 200 GET /images/main-bgfill.png (172.20.54.213) 9.95ms
4023 Nov 13 13:22:28 clc euca-console: INFO 200 GET /images/footer-bgfill.png (172.20.54.213) 2.84ms
4024 Nov 13 13:22:28 clc euca-console: INFO 200 GET /custom/Messages.properties?_=1384329140703 (172.20.54.213) 41.90ms
4025 Nov 13 13:22:28 clc euca-console: INFO 200 GET /custom/Messages_en.properties?_=1384329140779 (172.20.54.213) 10.44ms
4026 Nov 13 13:22:28 clc euca-console: INFO 200 GET /custom/Messages_en_US.properties?_=1384329140823 (172.20.54.213) 7.83ms
4027 Nov 13 13:22:28 clc euca-console: INFO 200 GET /fonts/play/Play-Bold-webfont.woff (172.20.54.213) 13.79ms
4028 Nov 13 13:22:28 clc euca-console: INFO 200 GET /images/css-sprites-small.png (172.20.54.213) 10.64ms
4029 Nov 13 14:08:53 clc snort[8336]: *** Caught Term-Signal
4030 Nov 13 14:08:53 clc kernel: device eth0 left promiscuous mode
```

Fig 11: Eucalyptus C and Python Components Logs

```
Nov 26 11:59:39 logserver anacron[4084]: Job `cron.daily' started
Nov 26 11:59:39 logserver anacron[4923]: Updated timestamp for job `cron.daily' to 2013-11-26
Nov 26 12:00:21 clc euca-cc: instances: 0000 (0000 extant + 0000 pending + 0000 terminated)
Nov 26 12:00:21 clc euca-cc:     nodes: 0001 (0000 busy + 0001 idle + 0000 unresponsive)
Nov 26 12:00:23 clc.main TRACE [Hmacv2LoginModule:New I/O server worker #2-14]
[com.eucalyptus.auth.login.Hmacv2LoginModule.makeSubjectString(Hmacv2LoginModule.java):141] VERSION2: POST#012172.20.54.211#012/services/
Eucalyptus/
#012AWSAccessKeyId=VYBSEIZHG6THUWVS9ATYJ&Action=DescribeAvailabilityZones&SignatureMethod=HmacSHA256&SignatureVersion=2&Timestamp=2013-11-
3A30%3A21Z&Version=2010-08-31&ZoneName.1=verbose
Nov 26 12:00:23 clc.main TRACE [Hmacv2LoginModule:New I/O server worker #2-14]
[com.eucalyptus.auth.login.Hmacv2LoginModule.makeSubjectString(Hmacv2LoginModule.java):141] VERSION2: POST#012172.20.54.211:8773#012/
services/Eucalyptus/
#012AWSAccessKeyId=VYBSEIZHG6THUWVS9ATYJ&Action=DescribeAvailabilityZones&SignatureMethod=HmacSHA256&SignatureVersion=2&Timestamp=2013-11-
3A30%3A21Z&Version=2010-08-31&ZoneName.1=verbose
Nov 26 12:00:30 clc.main TRACE [Hmacv2LoginModule:New I/O server worker #2-14]
[com.eucalyptus.auth.login.Hmacv2LoginModule.makeSubjectString(Hmacv2LoginModule.java):141] VERSION2: POST#012172.20.54.211#012/services/
Eucalyptus/
#012AWSAccessKeyId=VYBSEIZHG6THUWVS9ATYJ&Action=DescribeAvailabilityZones&SignatureMethod=HmacSHA256&SignatureVersion=2&Timestamp=2013-11-
3A30%3A28Z&Version=2010-08-31&ZoneName.1=verbose
Nov 26 12:00:30 clc.main TRACE [Hmacv2LoginModule:New I/O server worker #2-14]
[com.eucalyptus.auth.login.Hmacv2LoginModule.makeSubjectString(Hmacv2LoginModule.java):141] VERSION2: POST#012172.20.54.211:8773#012/
services/Eucalyptus/
#012AWSAccessKeyId=VYBSEIZHG6THUWVS9ATYJ&Action=DescribeAvailabilityZones&SignatureMethod=HmacSHA256&SignatureVersion=2&Timestamp=2013-11-
3A30%3A28Z&Version=2010-08-31&ZoneName.1=verbose
Nov 26 12:01:01 clc CROND[22051]: (root) CMD (run-parts /etc/cron.hourly)
Nov 26 12:01:01 clc run-parts(/etc/cron.hourly)[2205 starting 0anacron
Nov 26 12:01:01 clc run-parts(/etc/cron.hourly)[2206 finished 0anacron
Nov 26 12:01:01 nc1 CROND[29200]: (root) CMD (run-parts /etc/cron.hourly)
Nov 26 12:01:01 nc1 run-parts(/etc/cron.hourly)[2920 starting 0anacron
Nov 26 12:01:01 nc1 anacron[29211]: Anacron started on 2013-11-26
Nov 26 12:01:01 nc1 run-parts(/etc/cron.hourly)[2921 finished 0anacron
Nov 26 12:01:02 nc1 anacron[29211]: Will run job `cron.daily' in 38 min.
Nov 26 12:01:02 nc1 anacron[29211]: Will run job `cron.monthly' in 78 min.
```

Fig 12: Eucalyptus Java Components Logs

Thus, rSyslog used to give analysis of these aggregate logs. As seen before, to analyze these logs as a means of timeline approach, Awstats log parser with web based user interface has used as follows:

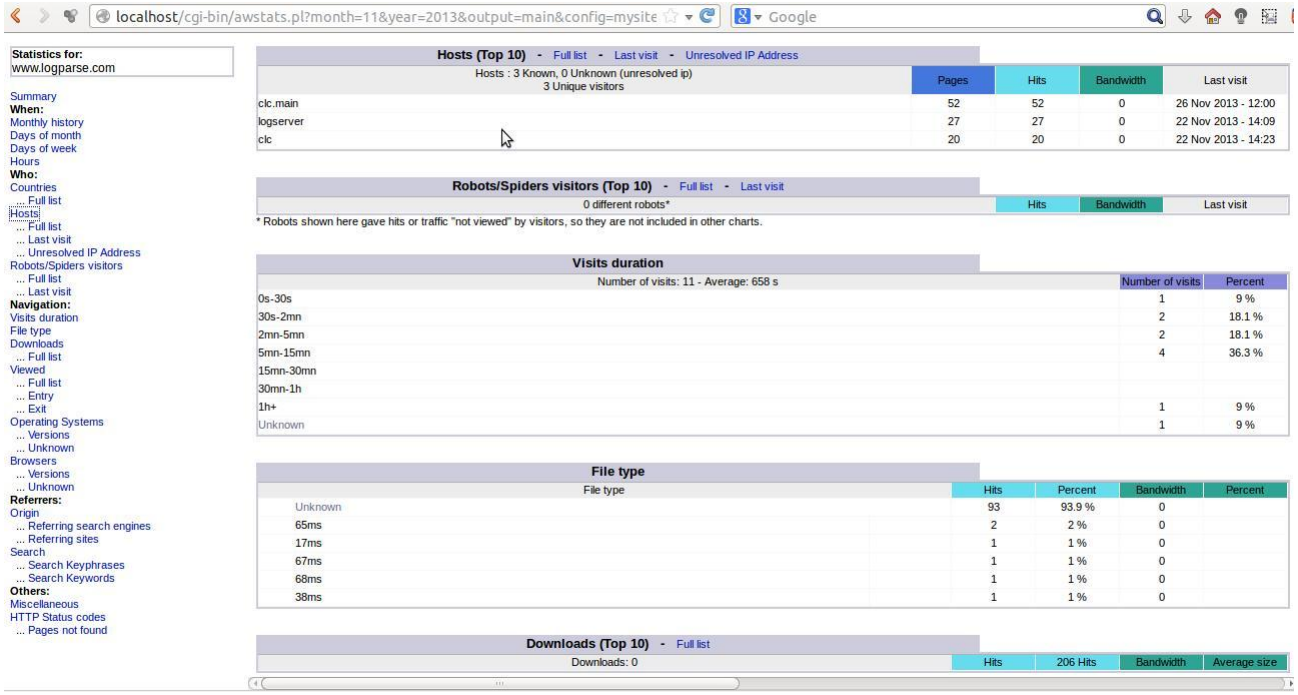

Fig 13: Parsing logs on rSyslog server with Awstats

The first three entries indicate names of three machines including hostname of rSyslog server. This shows timeline analysis approach simplifies the log analysis process.

4. EXPERIMENTAL OBSERVATIONS

From qualitative observation as well as literature survey, it has been found that Eucalyptus is more frontwards compared to Openstack. Three criteria are used to compare the experimental study.

- DDoS attack intensity
- Remote Log exportation
- IDS based Testbed

The table is as follows:

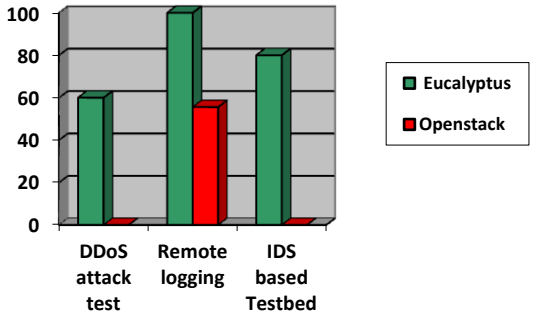

Fig 14: Graph based on Experimental Study

4.1 Criteria I: DDoS Attack Intensity

Eucalyptus has 5 components; CLC, W, CC, SC, NC.

CLC & Walrus are vulnerable to intrusions 3/5 = 60%

In Openstack, No prior testbed or implementation has observed = 0

4.2 Criteria II: Remote Logging

5 out of 5 components able to send logs to a remote server = 100%

4 out of 9 components able to send logs to a remote server = 55.56%

4.3 Criteria III: IDS based Testbed

IDS based testbed has implemented on CLC successfully.

5-1=4 4/5 = 80%

On the other hand, none prior work found in case of Openstack.

From this study, the conclusion is that the OpenStack is lagging behind in the area of Quality Assurance.

5. CONCLUSIONS

Summarize, the log analysis for Eucalyptus has been introduced in this scope of work. Eucalyptus is prone to attacks which are TCP SYN flood & SSL based DDoS. The expected outcome will be regarded Eucalyptus is that when the DDoS attacks on CLC has performed; it should be detected and monitored by Snort IDS. Whenever any suspicious activity occurs, snort should give alerts and logs of activity would get generated. This has been done by developing customized snort shared object rules. As stated previously in the survey, logs are spread across different devices in different format; by using rSyslog and configuring logging modules in the Eucalyptus, it is possible to export the cloud component logs from host network to the remote rSyslog server. The use of Awstats log analyzer makes easy to get results with minimal efforts but limited as unable to support variety of log formats parsing.

In short, Eucalyptus coming version will be able to export logs of its all components to the remote rSyslog server. Snort IDS shared object rules will detect intrusion attempts for attacking the system. The admin / developer is able to easily keep an eye on the events across the network.

6. FUTURE SCOPE

The scope of proposed work can be extended as follows: NC mostly comes in contact with virtual machines which are seemed too vulnerable to attack. Considering serious chances of intrusion detection, there is need of system like IDS packages embedded in each image interfacing with Eucalyptus so that any anomaly observed should take countermeasure actions as follows:

- Without shutdown image disconnect instance from the network.
- Stop the image (EBS based)
- Eucalyptus Logging for this image.

7. ACKNOWLEDGMENTS

We would like to thank the teams of Eucalyptus Development & Snort development for their great support to our work.

8. REFERENCES

[1] Eucalyptus,http://www.eucalyptus.com/eucalyptus-cloud/iaas/features

[2] Github, https://github.com/eucalyptus/eucalyptus

[3] Adeela Waqar, Asad Raza, Haider Abbas, 2011, User Privacy Issues in Eucalyptus: A Private Cloud Computing Environment" ,International Joint Conference of IEEE, Pg(s): 927-32.

[4] Zafarullah, Faiza Anwar, Zahid Anwar, 2011, "Digital Forensics For Eucalyptus",Frontiers of Information Technology, IEEE Computer Society, Pg. 110-16.

[5] Cloud Forensics: http://cloudtimes.org/2012/11/05/the-basics-of-cloud-forensics/

[6] Anna Sperotto, Michel Mandjes, Ramin Sadre, Pieter-Tjerk de Boer, and Aiko Pras, 2012, "Autonomic Parameter Tuning of Anomaly-Based IDSs: an SSH Case Study"IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, Pg. 128-41

[7] Jiong Zhang, Mohammad Zulkernine, and Anwar Haque, 2008, "Random-Forests-Based Network Intrusion Detection Systems" IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS—PART C: APPLICATIONS AND REVIEWS, Pg. 649-59.

[8] Chun-Jen Chung, Pankaj Khatkar, Tianyi Xing,Jeongkeun Lee and Dijiang Huang, 2013, "NICE: Network Intrusion Detection and Countermeasure Selection in Virtual Network Systems"IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, Pg. 198-211

[9] Jun-Ho Lee,Min-Woo Park, Jung-Ho Eom and TaiMyoung Chung, 2011, "Multilevel Intrusion

Detection System and Log Management in Cloud Computing", 13th International Conference on Advanced Communication Technology (ICACT), IEEE, Page(s): 552 - 555.

[10] Zhiyuan Tan, Aruna Jamdagni, Xiangjian He,Priyadarsi Nanda and Ren Ping Liu , 2013, "A System for Denial-of-Service Attack Detection Based on Multivariate Correlation Analysis", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, Pg. 1-11.

[11] Hannes Holm, 2013, "A large-scale study of the time required to compromise a computer system" IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, Pg. 1- 14

[12] Terrence V. Lillard, "Digital Forensics for Network, Internet, and Cloud Computing ", Syngress Publication Elsevier Inc.

[13] Denis Reilly, Chris Wren, Tom Berry, 2011, "Cloud Computing :Pros and Cons for Computer Forensic Investigations", International Journal Multimedia and Image Processing (IJMIP), Pg. 265 - 270.

[14] Hong Guo, Bo Jin, Ting Shang, 2012, "Forensic Investigations in Cloud Environments",IEEE International Conference on Computer Science and Information Processing(CSIP),Pg. 248 - 251.

[15] Stephen Biggs and Stilianos Vidalis, 2009, "Cloud Computing:The Impact on Digital Forensic Investigations",Copyright Institute of Electrical and Electronics Engineers, Inc, Pg. 1 - 6.

[16] Dominik Birk,Christoph Wegener, 2011, "Technical Issues of Forensic Investigations in Cloud Computing Environments",Systematic Approaches to Digital Forensic Engineering (SADFE), IEEE Conference Publications. Pg. 1 - 10

[17] George Sibiya,H.S. Venter,Sipho Ngobeni,Thomas Fogwill, 2012, "Guidelines for Procedures of a Harmonised Digital Forensic Process in Network Forensics",IEEE Conference on Information Security for South Africa (ISSA), Pg. 1-7.

[18] Luís Filipe da Cruz Nassif and Eduardo Raul Hruschka,2013,"Document Clustering for Forensic Analysis: An Approach for Improving Computer Inspection",IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, Pg. 46-54

[19] Hyun Jin Kim, Hong-Sik Kim, and Sungho Kang, 2011, "A Memory-Efficient Bit-Split Parallel String Matching Using Pattern Dividing for Intrusion Detection Systems", IEEE TRANSACTIONS ON PARALLEL AND DISTRIBUTED SYSTEMS, Pg.1904-11

[20] Meixing Le, Angelos Stavrou and Brent ByungHoon Kang, 2012," DoubleGuard: Detecting Intrusions in Multitier Web Applications",IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, Pg. 512-25

[21] Gideon Creech and Jiankun Hu,2013, "A Semantic Approach to Host-based Intrusion Detection Systems Using Contiguous and Discontiguous System Call Patterns" IEEE TRANSACTIONS ON COMPUTERS, Pg.1-14

[22] Francisco Maciá-Pérez, Francisco J. Mora-Gimeno, Diego Marcos-Jorquera, Juan Antonio Gil-Martínez-Abarca, Héctor Ramos-Morillo, and Iren Lorenzo-Fonseca, 2011,"Network Intrusion Detection System Embedded on a Smart Sensor" IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS, Pg. 722-32

[23] Mohsen Damshenas,Ali Dehghantanha,Ramlan Mahmoud, Solahuddin bin Shamsuddin, 2012,"Forensics Investigation Challenges in Cloud Computing Environments", 978-1-4672-1677-4,Cyber Security, Cyber Warfare and Digital Forensic (CyberSec), International Conference IEEE Conference Publications, Pg. 190 - 194.

[24] Brian Hay, Kara Nance, Matt Bishop, 2011,"Storms Clouds Rising: Security Challenges for IaaS Cloud Computing" ,Proceedings of the 44th Hawaii International Conference on System Sciences, Pg. 1-6.

[25] Stephen D. Wolthusen, 2009, "Overcast: Forensic Discovery in Cloud Environments", Fifth International Conference on IT Security Incident Management and IT Forensics, IEEE, Pg. 3-9.

[26] John Sammons,"The Basics of Digital Forensics:The Primer for Getting Started in Digital Forensics", Syngress Inc Elsevier Pub.

[27] Hping, https://en.wikipedia.org/wiki/Hping

[28] Wesley M. Eddy, Verizon Federal Network Systems, 2006, "Defenses Against TCP SYN Flooding Attacks", The Internet Protocol Journal - Volume 9, Number 4, December.

[29] SSL DoS, http://www.thc.org/thc-ssl-dos