International Journal of Engineering Research and General Science Volume 3, Issue 4, July-August, 2015
ISSN 2091-2730

A Practical Approach for the Detection of DDoS attack and It's Countermeasure selection in Open Stack using Xen Cloud Server

Sruthi.C, Sreyas.S, Archana.S.Narayanan, Anjaly Lakshmi.S, Jishnu.IJ
Department of Information Technology
Government Engineering College Sreekrishnapuram
Kerala, India.
Email: sruthicpalakkad@gmail.com
Contact No : +91 9496353377

Abstract— - Cloud Security is an attractive issue for research and development efforts. A cloud server means virtual server, which runs on a cloud computing environment. XenServer, is the best server virtualization platform, that can be used in world's largest clouds. OpenStack is a free and open source cloud computing platform. Xen Cloud Server can be obtained by integrating XenServer with OpenStack. Attackers can explore vulnerabilities of a cloud system and compromise virtual machines to deploy further large scale Distributed Denial of Service(DDoS). Common type of DDoS attacks include ICMP Flood, UDP Flood, SYN Flood. Snort is a widely used open source intrusion detection tool kit, that has to be installed on the cloud VM, which is the target of the attacker. It detects the attack and raises alerts. Appropriate countermeasures are selected from a set of countermeasures, thus establishing security in cloud.

Keywords— Cloud Security, Cloud Computing, Distributed Denial of Service, OpenStack, XenServer, Snort, Virtualization

INTRODUCTION

Users migrating to Cloud, consider security as the most important factor. Cloud Computing is the practice of using a network of remote servers hosted on the internet to store, manage and process data, rather than a local server or a personal computer[2]. The word cloud in cloud computing is used as a metaphor for the term "internet". So cloud computing means a type of internet-based computing, where different services such as servers, storage and applications are delivered to an organization's computers and devices through the internet. Attackers can exploit vulnerabilities in the cloud to deploy large scale DDoS attacks. In a DDoS attack, the attacker attempt to temporarily interrupt or suspend the services of a website, so that it is unavailable to users. Our system mainly focuses on DDoS attack detection and countermeasure selection in cloud.

XenServer is the best server virtualization platform, that is used in world's largest clouds. The cloud server, we are using is Xen Cloud Server. Xen Cloud Server is obtained as a result of integrating XenServer with OpenStack, an open source cloud computing environment. To give XenServer, a cloud infrastructure, we have used Devstack. Devstack is a bunch of scripts, that is used to quickly deploy OpenStack. OpenStack has five services: Nova(Compute service), Swift(storage service), Glance(image service), Keystone(Identity service), Horizon(UI service). We can create new users and launch instances in OpenStack. For giving DDoS attack to the target VM in cloud, we use metasploit framework in Kali linux. Kali Linux is a debian-derived linux distribution, designed for digital forensics and penetration testing. Metasploit framework is a tool for developing and executing exploit code against a remote target machine. It is a metasploit project's best known creation, a software platform for developing, testing and executing exploits. It can be used to create security testing tools and exploit modules and also as a penetration testing system.

For attack detection, a highly effective Network Intrusion Detection System(NIDS), called Snort is used. Snort monitors and analyzes the network traffic and raises alerts when an intruder intrudes. Snort can be configured in such a way that it can detect the DDoS attack of an attacker. We can customize the rules of Snort, edit the configuration file for desired performance. Snort is installed on the target machine(Target VM in cloud) of an attacker. By using ID generation algorithm, NAT algorithm, CMS algorithm and Marking algorithm, most severe attack is detected and countermeasure is selected from the pool of countermeasures. A virtual firewall can be

configured in the target machine, or standard countermeasures can be implemented for DDoS attacks, as a countermeasure. It focuses on a new approach of attack detection and countermeasure selection in Cloud

ARCHITECTURE

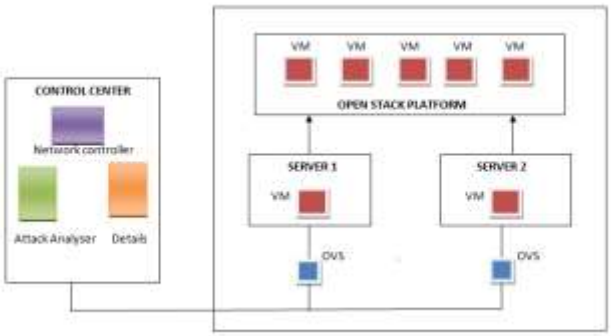

Figure.1 System Architecture

Main components in the architecture are

a. Cloud server
b. Virtual machines
c. Open Vswitch
d. Control center
    - A Network Controller
    - Details of Cloud Servers
    - Attack Analyzer

The architecture consists of two cloud servers that are integrated with OpenStack cloud platform. Cloud servers and OpenStack can create as many virtual machines as needed. Attack analyzer monitors the traffic in the network. It generates alerts depending on the types of the DDoS attack received and it also helps to obtain the most severe alert. The Network controller does the countermeasure selection based on the type of DDoS attack. All the information about the cloud server and VMs in it are present in Details of cloud server. Open Vswitch is a virtual switch used to interconnect the cloud servers and control center.

IMPLEMENTATION METHOD

Tools Used

XenServer, OpenStack, XenServer with OpenStack, Network Intrusion Detection Agent(Snort), Metasploit Framework (Kalilinux) are installed for the project.

a. XenServer

XenServer is the best server virtualization platform and hypervisor management platform that can be used in world's largest clouds. The Xen hypervisor, the heart of XenServer, is the most prolific public and private cloud hypervisor. XenServer is compatible with a variety of cloud management products. To make Xen Cloud Server, we have integrated XenServer with OpenStack cloud environment.

b. Openstack

OpenStack is a collection of Open source software project that can be collectively linked to operate a cloud network infrastructure in order to provide Infrastructure as a service(IaaS). OpenStack helps to deploy virtual machine and other instances which handle different tasks for managing a cloud environment on the fly.

c. XenServer with OpenStack

Integrating XenServer with OpenStack is a task that require a lot of resources and time. We use DevStack to deploy OpenStack[8]. DevStack is a bunch of scripts, used to quickly create an OpenStack development environment. We customize the local file according to our system configuration. The default service configured by DevStack are identity(Keystone), object storage(Swift), usage storage(glands), block storage(Cinder), compute(Nova), network(Nova), dashboard(Horizon), Orchestration(Heat).

d. Network Intrusion Detection System

Snort is used as network intrusion detection agent in a system[7]. Snort is an Open source intrusion detection tool kit for monitoring and analyzing the network traffic. When it identifies attack, it sends a report to the system. Snort works in three modes: Packet sniffer modes, packet logger mode, IDS mode.

• Sniffer mode:

Sniffer mode is the simplest Snort mode, and it is used to quickly capture the traffic.

• Packet logger mode:

Packet logger mode in Snort, logs the packet to disk.

• IDS mode:

Snort raises alert, when it detects a malicious activity in this IDS mode.

e. Metasploit framework

The metasploit frame work is an open source tool for developing and executing exploit code against a remote target machine[6]. Here metasploit framework is used to generate DDoS attack. It is setup in Kalilinux.

Algorithms Used

Four Different algorithms are developed for the project. The algorithms are ID generation algorithm, NAT algorithm, CMS algorithm and Marking algorithm.

a. ID generation algorithm

Input

• Alerts and priorities from snort database.

Output

• Generated ID, alert ID.

Algorithm

Step 1: Get the priority of each alert

Step 2: Set k=1 Step 3: Store the priority value in sorted order to an answerarray[]

Step 4: Get the value of alert id as k=k*10+1

Step 5: Repeat the step 4 for n times, where n is the number of alerts.

Step 6 Append the alert id to a retriarray[ ]

Step 7: Insert the value of the newly generated alert id on to the database.

b. NAT Algorithm

Input

• Different ID obtained from ID generation algorithm.

Output

• Alert ID of the most severe alert.

Algorithm

Step 1: Insert the first alert id onto the tree as the root node

Step 2: Get the next alert id

Step 3: If the length of the alert id is less than that of the root node, insert it as the left child of the root node using insert() function.

Step 4: If the length of the alert id is greater than that of the root, insert it as the right child of the root node using insert() function.

Step 5: Call the insert() function recursively on each node.

Step 6: Call the inorder traversal() function to get the value of nodes in ascending order.

Step 7: Store these values into an array.

Step 8: Last element in the array gives the most severe alert id

c. CMS algorithm

Input

• Type of attack obtained from the ID generation algorithm.

Output

• Best countermeasure for most severe attack.

Algorithm

Step 1: Find the alert of the more severe alert id.

Step 2: Find the type of attack corresponding to the alert from the database.

Step 3: Find the distance between the least intrusive alert and the most intrusive alert and store the value in the distance variable.

Step 4: Calculate the dummy value for each attack with different priority dummy value=cost * intrusiveness/priority.

Step 5: Find the countermeasure with priority 1 for the attack and whose dummy value is maximum.

Step 6: Return the countermeasure.

d. Marking Algorithm

Input

• Most severe alert, its attack type and best countermeasure, distance.

Output

• Status of each alert and specific work assigned to the alert.

Algorithm

Step 1: Select the countermeasure with maximum priority.

Step 2: Get the dummy value of the selected countermeasure.

Step 3: Find the mark for the countermeasure as Mark=dummy value * distance

Step 4: Enter the mark in the Status table.

Step 5: Set the corresponding status of the alert as ' Examined '.

ANALYSIS

In this chapter analysis of Snort is provided. Analysis is done on the basis

of types of traffic analyzed, alert generation in different networks and packet loss

in host and virtual environment.

Types of Traffic Analyzed

When the Network Traffic was analyzed for 2 minutes TCP, UDP, ICMP and Synflood alerts were received. Out of 2893 alerts raised by Snort 1157 were TCP alerts, 723 alerts were of UDP, 578 alerts were of type ICMP, 434 Synflood alerts. Based on this percentage of each alerts were calculated as shown in table. A pie chart is constructed taking these values from the table I. From pie chart, we analyzed that majority of traffic analyzed by snort is of type TCP. least alerts were generated for Synflood attack.

TABLE I. TYPES OF TRAFFIC ANALYSED IN 2 MINUTES

TCP	40
UDP	25
ICMP	20
SYN FLOOD	15

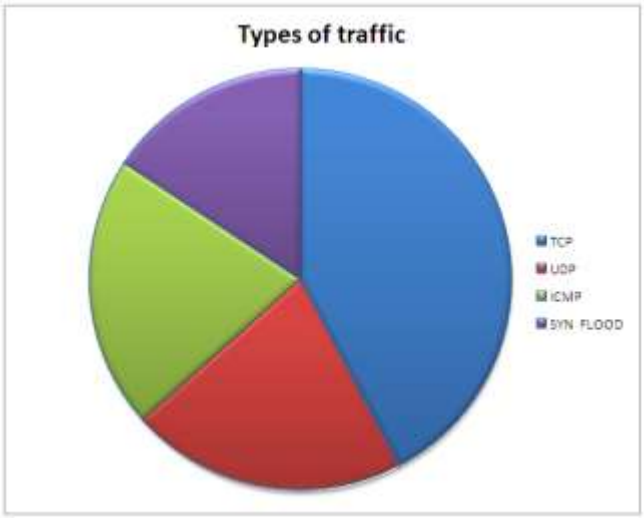

Figure.2 Types of traffic analyzed

Alert Generation in different Networks

When the alerts raised by Snort were observed under different network speed. All packets were monitored by Snort working at a speed of 54 Kbps. So alert generation is 100 percentage. When we examined the Snort under 54 Kbps, 152 Kbps, 550 Kbps, 720 Kbps and 1000 Kbps, results obtained are shown in the table II. We can analyze that as the network speed increases there is a slight reduction in the alert generation percentage. This is shown in figure 3.

TABLE II. ALERT GENERATION

Network Traffic (Kbps)	Alert generation in percentage
54	100
152	100
550	90
720	84
1000	72

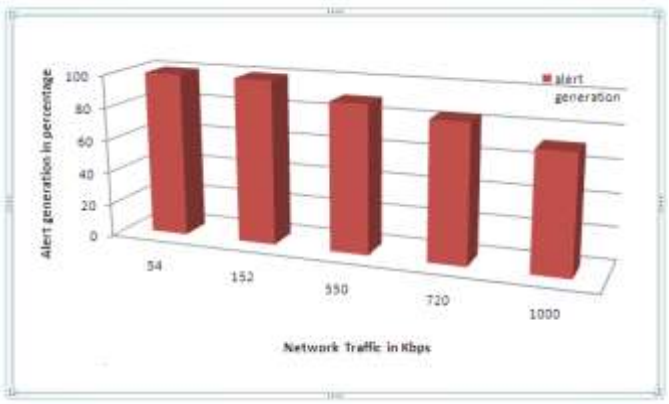

Figure.3 Alert generation

Packet loss in Host and Virtual environment in different Networks

When the Network Traffic was monitored by Snort in host and virtual environment under different network speeds, the table III was obtained. It was analyzed that at 54 Kbps, both host and virtual environment showed zero percentage packet loss. when the network speed was again increased to 550 Kbps a packet loss of 4 percent was seen in host environment, and a packet loss of 1 percent was seen in virtual environment. When this process was repeated for a higher network speed of 720 Kbps, 1000 Kbps packet loss increased in both host and virtual environment. It was analyzed that packet loss in virtual environment is less when compared to host environment.

TABLE III. PACKET LOSS IN HOST AND VIRTUAL ENVIRONMENT

Network Speed(Kbps)	Host	Virtual
54	0	0
152	1	0
550	4	1
720	8	3
1000	13	5

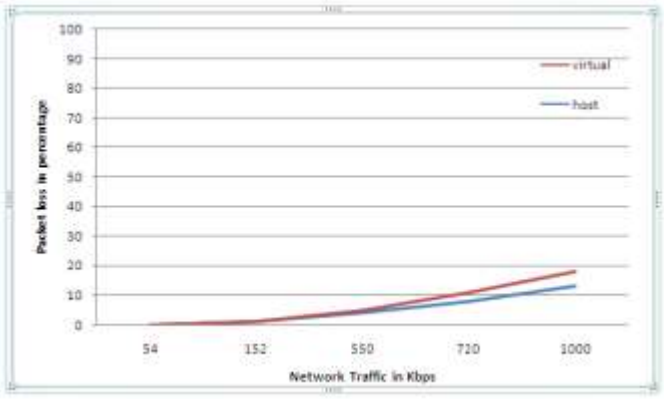

Figure.4 Packet loss in host and virtual environment

PERFORMANCE

XenServer - XENISM

Out of 8079 MB XenServer uses 967 MB and DevStack uses 4096 MB. 1700 MB RAM is allocated to Ubuntu VM running on it. XenServer performs smoothly under this memory condition. 0.631 of the total memory is used by XenServer. These are shown in the figure 5 given below.

Figure.5 Memory Utilization

OpenStack

The figure 6 shows the Hypervisor summary of OpenStack. The hypervisor used is the XenServer shown in the figure 'XENISM'. The details of XenServer can be viewed in OpenStack. The fig shows that the memory usage of XenServer on OpenStack is 2.8 GB of 39.4 GB. The CPU usage is 50 percent (5 out of 10).

Figure.6 Hypervisor summary

Snort - RAM Utilization

RAM utilization of Snort is given in the table IV. It shows the performance of Snort in a time interval of 30 seconds. On an average the RAM used by Snort is 0.6875 GB. This is formed by taking average of values obtained from table IV. A line graph is plotted from these values and it was observed that the RAM usage of Snort is minimum and constant, which results in its good performance.

TABLE IV. RAM UTILIZATION

Time (S)	Ram Utilization
30	0.603
60	0.665
90	0.721
120	0.761

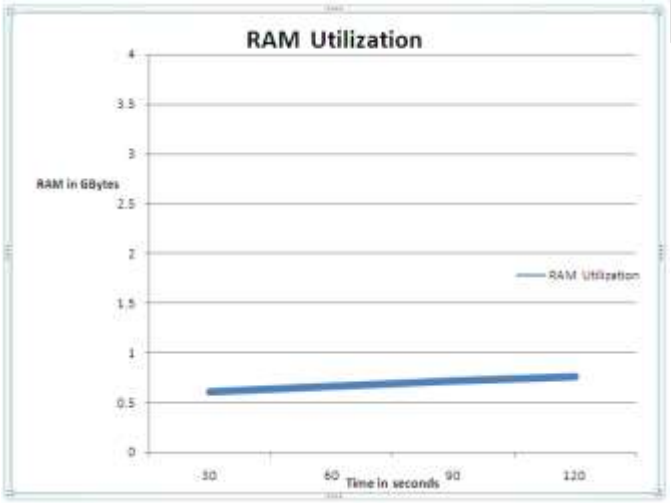

Figure.7 RAM utilization

Overall Performance

The overall performance of the whole system is measured in terms of :

• CPU performance

• Memory performance

• Network performance

a. CPU Performance

The CPU performance graph given in figure 8 shows that the system uses the CPU 1 and CPU 2 equally. At the time 1.42 PM the CPU utilization is maximum while at 1.38 PM the CPU utilization is minimum.

Figure.8 CPU Performance

b. Memory Performance

Initially the RAM used is near to 0 GB but as the system starts functioning this value is nearly 6.5 GB. From the figure 9, the memory performance of the system is constant.

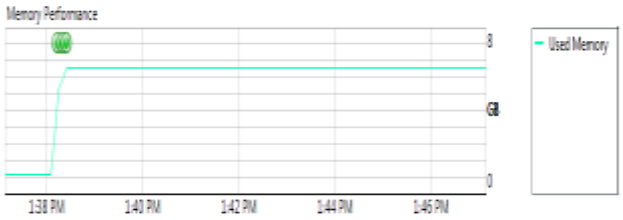

Figure.9 Memory Performance

c. Network Performance

Network performance of the system is monitored in 90 Kbps network. The network performance changes as the network traffic increases. Network traffic of the system is shown in the figure 10.

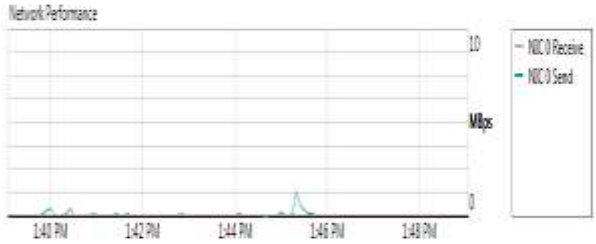

Figure.10 Network Performance

CONCLUSION

Organizations that operate or use Internet connected services such as websites, portals and Cloud services need to be aware of threats that can disrupt service. Our System focuses uncommon attack vector that has been used to attempt to adversely affect Cloud services on the Internet, DDoS. The most common types of DDoS attacks include SYNflood, DNS amplification, malformed TCP and UDP packets. We use the metasploit framework in Kali Linux for giving attacks to cloud server. The system makes use of a widely used intrusion detection tool kit, Snort for detecting the attacks to cloud. It is a highly effective tool, for network intrusion detection. Appropriate Countermeasures are selected from a pool of countermeasures. Thus, the system aims at attack detection and countermeasure selection in cloud services.

The Future work of the project is to configure a virtual firewall on the OpenStack cloud server. Developing a good false alarm detecting algorithm is also considered as a future work. The entire system can also be implemented on host based IDS, so that a comparative study can be made between current system and host based system.

REFERENCES:

[1] Chun-Jen Chung, Pankaj Khatkar, Tianyi Xing, Jeongkeun Lee and Dijiang Huang, "NICE: Network Intrusion Detection and Countermeasure Selection in Virtual Network Systems", *IEEE Transaction on dependable and secure computing* Vol: 10, No:4, Year 2013

[2] Z. Duan, P. Chen, F. Sanchez, Y. Dong, M. Stephenson, and J.Barker, "Detecting spam zombies by monitoring outgoing messages," *IEEE Trans. Dependable and Secure Computing*, vol. 9, no. 2, pp. 198–210, Apr. 2012.

[3] "Openflow", *http://www.openflow.org/wp/learnmore/*, 2015.

[4] Mitre Corporation, "Common vulnerabilities and exposures,

CVE" *http://cve.mitre.org/*, 2015.

[5] O. Database, "Open source vulnerability database (OVSDB)",

http://osvdb.org/, 2015.

[6] "Metasploit ", *http://www.metasploit.com*, 2015.

[7] "Snort" *http://www.snort.org*, 2015.

[8] "XenServer " *http://www.xenserver.org*, 2015