Usage of Honeypot to Secure datacenter in Infrastructure as a Service data

Ms. Priyanka Paliwal
M. Tech. Student 2nd yr.(Comp. Science& Eng.)
Government Engineering College Ajmer
Ajmer, India
(Erpriyanka_paliwal06@rediffmail.com)

Mrs. Prakriti Trivedi
Assistant Professor (CSE/IT)
Government Engineering College Ajmer
Ajmer, India
(hod_ceit@rediffmail.com)

Abstract—This paper is about deploying Honeyd, a virtual honeypot that simulates different operating system on Eucalyptus Iaas cloud and usage of proposed encryption methodology on data to make data access secure. The data collected in capture mode is analyzed to prevent further exploitation of system.

Keywords- Honeypot; Honeyd; Virtual Honeypot; Iaas; Low interaction honeypot

I. INTRODUCTION

A honeypot is an instrument for information gathering and learning. The focus lies on a silent collection of information about attack patterns, used programs, purpose of attack and the black hat community itself to learn about their motives, technical knowledge and abilities. We categorize types of honeypots based on the level of interaction they offer to the attackers as:

1) Low-interaction honeypots

2) Medium-interaction honeypots

3) High-interaction honeypots

Low-interaction honeypots simulate only some parts of the system as the network stack. Low-interaction honeypots simulate only services that can't be used by an attacker to get full access to the honeypot and thus are not able to control the system. Use of these honeypots includes identification of port scans, generation of attack signature and malware collection. Popular examples of this kind of honeypots are Honeyd and Nepenthes, Specter, and KFSensor. They are also used for analyzing spammers or detecting worms [1].

A high-interaction honeypot is a conventional computer system or a fully functional Virtual Machine, a router or a switch. Here attackers can interact with a real system where almost nothing is restricted and thus, it is more risky compared to low-interaction honeypots. Therefore this kind of honeypot is normally placed behind a firewall to lessen the risk. They are not easily deployable compared to low-interaction honeypots, but by using them we learn more about the attackers behavior and find new vulnerabilities. Argos, Potemkin and Honeybow are some high interaction honeypots.

Honeypots are also classified as:

1) Physical honeypots

2) Virtual Honeypots

A physical honeypot is a real machine on the network with its own IP address. This is often high-interactive so allow the system to be compromised completely if compromised. This being expensive to install and maintain it is impractical for large address spaces to deploy a physical honeypot for each IP address so we prefer virtual honeypots.

A virtual honeypot is simulated by another machine that responds to network traffic sent to the virtual honeypot and responds to the traffic sent to it. A virtual Honeypot may simulate a lot of different virtual honeypots at the same time.

They include Honeyd and Kfsensor . Niels Provos created the original Honeyd (honeypot daemon) in 2002 as an open-source UNIX tool. Michael A. Davis, released a ported version of Honeyd in March 2003. Recently open-source Honeyd is become available for Windows [2].

A "cloud" is a set of machines and web services that implement cloud computing. Cloud computing is the access to computers and their functionality via Internet or a local area network. Users of a cloud request this access from a set of web services that manage a pool of computing resources (machines, network, storage, operating

systems, application development environments, application programs). The cloud can expand and contract. The elasticity means that the user can request additional resources on demand and just as easily release those resources whenever no longer needed. Elasticity is the main reason why people are moving towards cloud. When granted, a fraction of the resources in the pool is dedicated to the requesting user until he or she releases them. Eucalyptus Cloud uses virtual machines and physical systems as nodes in clusters.

A virtual machine (VM) is a software implementation of a machine (i.e., a computer) that executes programs like a physical machine. Each VM includes its own kernel, operating system, supporting libraries and applications. A hypervisor provides a uniform abstraction of the underlying physical machine. Multiple VMs can execute simultaneously on a single hypervisor.

Cloud is classified as:

1. Infrastructure as a Service (IaaS) cloud

IaaS clouds provide access to collections of virtualized computer hardware resources, including machines, network, and storage. With IaaS, we assemble our own virtual cluster on which we install, maintain, and execute our own software stack. We have used IaaS cloud in our paper as we had options to install different cloud images suiting our VM operating system.

2. Platform as a Service (PaaS) cloud

PaaS style clouds provide access to a programming or runtime environment with scalable compute and data structures embedded in it. With PaaS, we develop and execute our own applications within an environment offered by the service provider.

3. Software as a Service (SaaS)

SaaS style clouds deliver access to collections of software application programs. SaaS providers offer users access to specific application programs controlled and executed on the provider's infrastructure. SaaS is also referred as "Software on Demand [3]."

II DEPLOYMENT OF HONEYD ON IAAS DATA

Eucalyptus is a Linux-based open-source software architecture that implements efficiency-enhancing private and hybrid clouds within an enterprise's existing IT infrastructure. Eucalyptus is an acronym for "Elastic Utility Computing Architecture for Linking Your Programs to Useful Systems."

Private clouds give us immediate access to computing resources hosted within an organization's infrastructure. Our self-provision and scale collections of resources make it suitable to be deployed either on a single system with virtual nodes and thus reduces expenses or on a massive network. Because it is deployed within the organization's existing data center and behind the organization's firewall a private cloud is subject to the organization's physical, electronic, and procedural security measures and thus offers a higher degree of security over sensitive code and data.

Eucalyptus is deployed without modification on all major Linux OS distributions, including Ubuntu, RHEL, Centos, and Debian. Ubuntu distributions now include the Eucalyptus software core as the key component of the Ubuntu Enterprise Cloud [4].

We have used Ubuntu 10.04 LTS to install Eucalyptus v2.0.3. Eucalyptus Open Source v2.0.3 Eucalyptus Machine Image (EMI) is pre-configured operating system and virtual application software that can be used to create an instance in either Amazon's EC2 environment or a Eucalyptus environment.

The following are EMIs provided by Eucalyptus to help get users started with using Eucalyptus "Fig. 1".We are provided with versions of each EMI - "default" which has a 1.3 Gig root file system, and "large" which has a 4.5 Gig root file system.

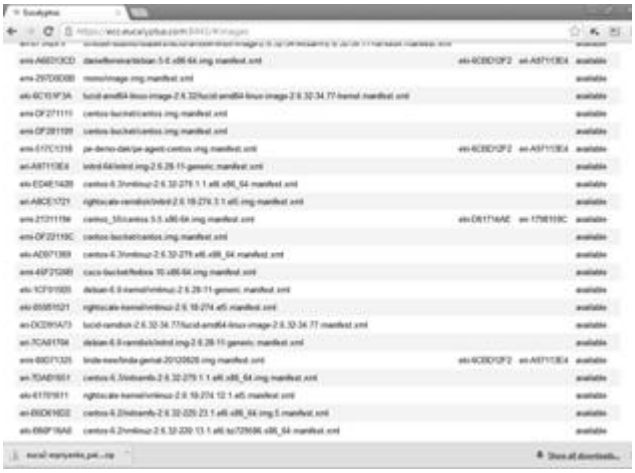


Figure 1 - EMIs provided by Eucalyptus v2.0.3

When using the *"large"* image type, we must make sure that the correct virtual machine type is being used when launching an instance using the image.

Additionally, each image has a custom startup script that allows the utilization of bash scripts to be passed by the 'user-data' option with the euca-run-instance command and executed on instance startup. Once we have downloaded an image, we bundle, upload and register it for use in our Eucalyptus cloud "Fig 2".

Figure 2- Eucalyptus image configuration screenshot

III HONEYD LOGGING

Honeyd framework has capability to provide different ways of logging network traffic. Honeyd supports features that make the daemon flexible for creating both host based and network based virtual honeypots. It Simulates thousands of virtual hosts at the same time and operating systems at TCP/IP stack level having arbitrary routing topologies. It is used to create multiple virtual honeypots on a single machine. We have installed Honeyd to run a range of services like FTP, HTTP, or SMTP on window based virtual machine node on IAAS cloud cluster . Honeyd logging provides packet level logging with (-l) command line option, and service level logging with (-s) for running honeyd. Honeyd has packet-level log files and service-level log files. Packet-level log file contains source IP address and port, destination IP address and port, the timestamp when a packet was received, and which protocols and ports are being used. If the connection is established, then honeyd logs the information about the start and end time of the connection, and transmitted bytes. Below "Fig.3" is the data from packet-level log file.

The first field displays the time to receive packet. The second field contains the protocol information- tcp and icmp. Next field displays connection type which either may be *S* (start of new connection), *E* (the end of the connection) or connection neither S nor E. Next fields represent source IP address, source port of the packets, and destination IP address, destination port of the virtual honeypots. The last fields represent transmitted bytes sent and received by Honeyd, and type of operating system .

Service level logs gives us detailed information about incoming traffic and are based on the output of the emulated scripts like web, telnet, IIS, PWS etc.

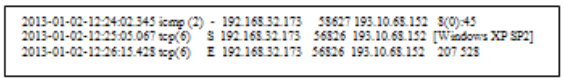


Figure3: output from packet-level log file

IV STEPS TO CONFIGURE HONEYD

After downloading Honeyd, define the IP addresses of cloud controller (CLC) on which the software listen one or more virtual operating systems. We have installed Ubuntu 10.04 LTS , Ubuntu 12.04 LTS, Windows XP and Windows 7 as virtual machines on cluster. The Honeyd uses IP ports logging options which are mentioned as command-line parameters in the configuration file. When Honeyd starts, it immediately looks for a configuration file in which we have defined virtual host templates, bound IP addresses to those templates, specified ports and services. Templates are Honeyd's method for keeping track of one or more virtual OS environments. Each template defines a unique IP address associated with a virtual operating systems ports, and services. One instance of Honeyd can support many templates. Every configuration file contains a template called default which is the template Honeyd uses when no other template applies.

We define a template as: Create <templatename>

We can configure Honeyd to use specific, predefined IP addresses or a range of addresses. Other options are to have Honeyd respond to any request for a currently invalid address or to any packet it sees on the wire. On the command line, define the subnet Honeyd emulated IP as:

honeyd 192.168.169.1-192.168.169.255

We have not published the original emulated IP address range of Honeynet as hackers and worms constantly scanning across the Internet for machines could exploit our Honeynet.

To bind a Honeyd template to a specific IP address, we used the configuration file statement : bind <ipaddress><templatename> [5]

Template name in our cluster can be either of Win7, WinXP. Although configuring honeyd from Windows GUI is not possible henceforth it needs UNIX emulator to get existing scripts work. The Data Analysis is used for analyzing real-time flows of incoming and outgoing data. Alerts can be taken by *whois* capability per day. It is possible to download the packet data in pcap format. We examine all connections per day, and most connected destination and source IP addresses and ports, information about protocol type, number and bytes of the packets, and OS type of source IP address of the connection.

Traffic going to and from Honeyd provided us with a roughly 1,50,000 KB PCap file recorded in a week's time "Fig. 4". This record has all our intruder's moves, from the time of the intrusion to the use of our node. We found that scans that occurred across our system on the Honeyd Iaas data are most likely worm type.

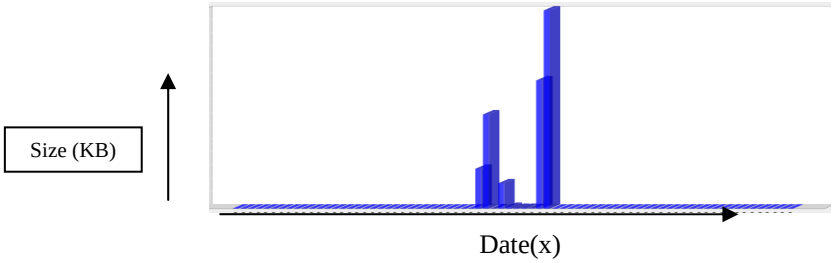
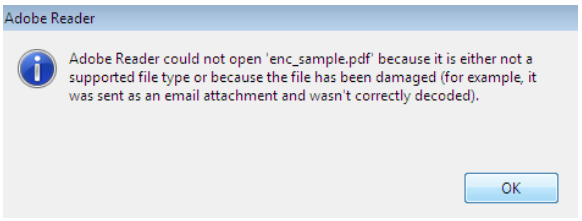


Figure 4: Pcap binaries obtained based on date(x) 02/01/2013 08/01/2013 and size in kb (0 to 1,50,000)(y)

V PROPOSED ENCRYPTION ALGORITHM AND RESULTS

We have used symmetric secret key cryptography as the same key is used for encryption as well as decryption. The sender uses the passkey to encrypt the plaintext and sends the cipher text to the receiver. The receiver applies the same passkey to decrypt the message and recover the plaintext. With this form of cryptography, it is obvious that the key must be known to both the sender and the receiver but once the data is exchanged the key could be randomly changed i.e. once the file is exchanged the next time it could be encrypted using the different passkey. For decryption the same passkey is to be used else it results into data not being decrypted.

The figure below Fig. 5 shows the message generated when the user attempts to open encrypted file 'enc_sample.pdf' without correct passkey or without any passkey.



The following constraints are used while implementing the algorithm. The password is supposed to have minimum 6 to 8 characters and supports uppercase, lowercase or special characters. If password is not in range specified the data is not encrypted and asks for the same. This algorithm could be implemented on any type of file format and adds a prefix to the file name which is encrypted.
Encryption and decryption both processes are implemented on the entire file. The table below Table1 shows the file types, encryption and decryption time taken with the proposed algorithm hence forth:

Table 1

File Name	File Size	Encryption Time	Decryption Time
Sample1.pdf	303 kb	31ms	11ms
Era.flv	60928kb	1342 ms	703ms
Waterfall.jpg	132 kb	16ms	0 ms
Tictactoe.pdf	86kb	16ms	16ms

Data Collected with proposed Algorithm

The next figure Fig. 6 shows the Matlab simulation of the proposed algorithm. The Red line depicts the encryption time taken and blue line indicates the decryption for the files correspondingly. The file with encryption time 1342 ms took 703ms for decryption. The above mentioned table1 data plots are shown in Fig. 6

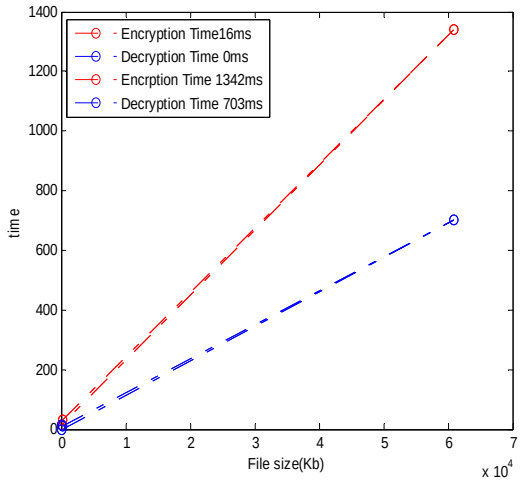


Figure 6: Matlab Simulation of the Proposed Algorithm

VI Conclusion

We installed Honeyd on Eucalyptus Iaas cloud cluster and captured data about the exploits on the system. The proposed encryption algorithm is applied on files in cloud, using symmetric key cryptography concept. This being low interaction honeypot is more exposed to compromises. The data captured is analyzed daily so that attacks are found and further access by the intruder be prevented. The levels of security provided by using Eucalyptus cloud and virtual operating systems along with deployment of virtual honeynet 'honeyd' along with the encrypted files using the proposed algorithm makes the data unassailable.

VI References

[1] http://www.projecthoneypot.org
[2] http://www.honeyd.org/contrib.php
[3] http://www.eucalyptus.com
[4] http://www.eucalyptus.com/docs/3.2/ig/installing_euca2ools_standalone.html
[5] http://www.windowsitpro.com