International Journal of Information and Computation Technology (IJICT)

ISSN 0974-2239 Volume 3, Number 10 (2013)

EMULATION OF FTP SERVICE USING VIRTUAL HONEYPOT

Aastha Bhandari[1], Sanmeet Kaur[2]

[1,2]School of Mathematics and Computer Applications, Thapar University,
Patiala, Punjab, INDIA.

ABSTRACT

Security, over the last decade, has become a huge priority for the network administrators. They dedicate lots of time to make sure that each network has the best and latest security patches, firewalls, and intrusion detection systems. Unfortunately, the security patches such as firewalls and intrusion detection systems are not as effective as they used to be due to the generation of large log files. Both the above mentioned techniques have information overload problems which can be solved using Honeypots. Honeypots interact with the attacker and collect the data that is thereafter analyzed. Honeypots are resources which are targeted by attackers and upon attack it logs data about the attacks. FTP server can be emulated on the honeypot and various scans can be done on the emulated service. The ip addresses trying to perform some malicious activity on ftp service can be caught and their data can be logged in the log file. Thus honeypot is a very good tool to find the vulnerabilities in the security system which are used by intruders.

Keywords- Virtual Honeypot; Honeyd; FTP Emulation.

1. INTRODUCTION

Honeypot systems are decoy servers or systems that have value in being attacked or probed. The purpose of honeypot system is to lure attackers and study their techniques and tools. These systems are made purposely insecure so that attackers can find them attractive. On production systems it is difficult to detect attacks among so much legitimate traffic. So honeypots are useful because anything destined to and coming from it is considered malicious. Honeypots are generally implemented on the top of an operative system using its networking capabilities, TCP/IP stack[5].

There are two types of honeypot: Low interaction and High Interaction honeypots based on the information that lies within the honeypot[2]. Low interaction Honeypots are those which do not contain any production value. These honeypots are used merely for confusing the attacker and buying some time from the attacker. Some examples of low interaction honeypots are Specter, Honeyd, KFSensor.

On the other hand, High interaction honeypots are those which have some valuable information for the attacker which is of real importance. Such honeypots are used to study the patterns, techniques and tools used by attackers. High-interaction honeypots let the hacker interact with the system as they would any regular operating system, with the goal of capturing the maximum amount of information on the attacker's techniques. Some of the examples are Mantrap and Honeynet.

In this paper a low-interaction honeypot is implemented as a software process that can emulate the behavior of different operating systems and network services is defined, deployed and the resulting traffic is analyzed.

2. LOW INTERACTION HONEYPOTS

Low-interaction honeypots present the hacker a set emulated services with a limited subset of the functionality they would expect from a server, with the intent of detecting sources of unauthorized activity.

2.1 Introduction

Because of their simple design and basic functionality, Low interaction honeypots are the easiest to install, configure, deploy and maintain. Generally such technologies merely emulate a variety of services. The attacker can interact with the limited predefined services. For example, a low interaction honeypot can emulate a standard Linux or Windows server with several running services like FTP, Telnet etc. An attacker can send FTP request to

the honeypot, get banner that states the operating system and perhaps obtain a login prompt. The attacker can try to login to the FTP server by using brute force or by guessing the passwords. The honeypot would capture and collect those attempts, but there is no real operating system for the attacker to log on to.

Low interaction honeypots are simple, thus they have the lowest level of risk because less functionality offered leads to less mistakes. As there is no real operating system for the attacker to interact with thus attacker cannot use honeypot to attack and monitor other systems. there are many low level interaction honeypots available in market like BOF, Specter, Honeyd etc.

Disadvantages of low-interaction honeypots originate from the same properties as do the advantages. While we can emulate the responses an attacker would expect for known vulnerabilities and exploits, a low interaction honeypot may not respond accurately to exploits that have not been expressed for emulated services[3]. This limits the honeypot's ability to aid in discovering new vulnerabilities that are being exploited or new attack patterns that can utilize those vulnerabilities.

2.2 Overview of BackOfficer Friendly(BOF), Specter and Honeyd

2.2.1 BOF BackOfficer Friendly was developed by Marcus Ranym in 1998. BOF is a low interaction honeypot which can run on Windows or Unix operating systems[2]. It can monitor up to seven emulated services. BOF is very simple, flexible and free. But it offers little interaction with the attacked such that false banners or bogus information like password files cannot be given to attacker to lure the attacker because it does not have such capabilities.

2.2.2 Specter Specter is a smart honeypot-based intrusion detection system[4] Specter is a commercial honeypot created and supported by NetSec. Specter works only for some Windows systems. A software solution is installed on a system and emulates a variety of services attackers can interact with which is limited to whatever functionality the specter software provides. Specter have the capability of emulating not only the interactions but vulnerabilities also. It can emulate 13 services. It can imitate like the real world applications by responding with Fake Replies which are more realistic than BOF . The problem with Specter is it operates at the application level which means IP stack is not emulated and it can only emulates the chosen OS based on the seven emulated services. For the tools like Nmap it is easy to detect Specter honeypot which can cause problem.

2.2.3 Honeyd Honeyd is developed by Niels Provos. It is a low-interaction honeypot. Honeyd monitors the unused IP address space[1] within a local area network it is installed in. It can only emulate services and is used to detect attacks or unauthorized activity. It is an Open Source solution. It has two advantages, one is it can detect connections on TCP port and the other is the emulated services can be modified and the level of interaction as well.

Various low interaction honeypots are compared in the tabular form as follows:

Table 1. *Comparison of various low interaction honeypots*

	BOF	**Specter**	**Honeyd**
Freely available	No	No	Yes
Open source	No	No	Yes
Log file support	No	Yes	Yes
OS Emulation	No	Yes	Yes
Services Emulation	7	13	Unrestricted

2.3 How Honeyd Works

Honeyd can emulate numerous services running on different operating systems like FTP server on Windows or Linux, Telnet on linux etc. Honeyd works on the principle that when it receives a probe or a connection for an emulated service, it assumes that the connection is hostile and most likely a probe, scan or an attack. When honeyd receives such traffic, it assumes that the IP address of the source is a potential attacker. It then starts the emulated service for the port and once the emulated service is started, it interacts with the attacker and captures all the activity in a log file. When the attacker is done, the emulated service exits and stops running. Honeyd then continues to wait for any more traffic or connection attempts to systems that do not exist. Honeyd assumes an IP address and emulates the service only when it receives a connection attempted this is an extremely efficient method.

3. IMPLEMENTATION

To gather the information on the network about the type and source of attacks, a low interaction honeypot Honeyd was implemented. Following steps were taken (from scratch) to emulate ftp server on Honeyd:

3.1 Setup Honeyd on VMware Workstation

VMware Workstation is a hypervisor which enables users to set up multiple virtual machines (VMs) and use them simultaneously along with the actual machine. Each virtual machine can execute its own operating system, such as Microsoft Windows, Linux or BSD variants. As such, VMware Workstation allows one physical machine to run multiple operating systems simultaneously.

3.1.1Software Used

1. VMware Workstation (Using VMware Workstation 6)
2. Any Linux platform.(For this experiment, using Ubuntu 12.04)
3. Honeyd (http://www.honeyd.org/release.php using version 1.5c)

3.1.2 Hardware Used

1. A computer running Microsoft Windows OS (Using Windows 7)

3.2 Installing and Configuring Honeyd

As an OpenSource solution designed for Unix, Honeyd does not provide any support or a nice GUI for installation or configuring.

3.2.1 Steps of Installation

1. Download the source code of honeypot: In linux one can install honeyd using the following command

   #apt-get install honeyd honeyd-common

2. Compile the source code
3. Install the Honeyd binary and configuration files.
4. Run the Honeyd binary from the command line, using the two configuration files, to the network we want it to monitor.

Once started, Honeyd will interact with any attack sent its way. The command to start Honeyd is

honeyd -p /etc/honeyd/nmap.prints -f /etc /honeyd /honeyd .conf

3.3 Using farpd

It is impractical to and resource consuming to have a single honeypot claim traffic for a group of IP addresses with means of configuring its network interface for all the ARP requests arriving at the network interface of the honeypot. ARP requests are broadcast packets used to discover which machine (more specifically which MAC address) has a specific IP address. Under normal circumstances the operating system takes care of responding to ARP requests. Farpd has the ability to reply to these requests, the traffic can be effectively directed for any IPs in the LAN to the required host, without actually configuring the host network interface for all these addresses[2]. Farpd is used as follows:
 # farpd <ip address>

3.4 Assumptions

The following information are used in the deploying environment but settings can be changed according to the experiment:

Table 2. *Assumptions about IP addresses*

IP address of my Windows (Host OS) computer	172.31.28.247
IP address of Ubuntu(Guest OS) machine in VMWare	172.31.28.131
IP address of Honeyd emulated FTP server	172.31.28.244

3.5 Emulation of FTP service on Honeyd

The default personality of the honeypot was set to emulate a Microsoft Windows Server 2003 system. After this port 21 was opened and set to execute “ftp.sh” shell script to emulate a ftp session. Since the honeypot exposes only port 21 rest of the ports were set to reset this action emulates the behavior of a closed port. The configuration used in the experiment is given below:

create ftptemplate

set ftptemplate personality "Microsoft Windows Server 2003"

set ftptemplate default tcp action reset

set ftptemplate default udp action reset

set ftptemplate default icmp action open

add ftptemplate tcp port 21 "/usr /share/honeyd/scripts/unix /linux/ftp.sh"

bind 172.31.28.244 ftptemplate

Above statements were written in honeyd.conf file, the ftp.sh file comes with honeyd installation and resides under “/usr/local/honeyd/scripts/unix/linux” directory path. The name ftptemplate was given to the ftp personality and bound with the IP 172.31.28.117 . Whenever anyone tries to make a connection with the given IP address it'll get an effect of connecting to ftp server when actually there is no ftp server related to this IP address.

4. RESULT AND CONCLUSION

After emulating the ftp service, various scans were done on the ftp emulated honeypot like ping, nmap, tracert. Tracert, ping and nmap gave good results that the port number 21 is open (which was the main goal of the emulation). Nmap was important because most of the attackers run nmap for OS fingerprinting and checking the open ports. Ftp server asked for the username and password after the connection between the host machine and ftp server was established. All other usernames except anonymous were invalid as per the script. When higher ports were opened in the ftp emulated service some results that came into view are worth considering. There were many ip addresses that were trying to connect to those unregistered ports but as our service was instructed to kill all such connections, it killed. The presented paper can be seen as initial work for implementing the collaboration of honeyd with other intrusion detection system. The log file generated by honeyd can be fed into IDS which can help it in making new rules for new threats whose signatures are not yet identified.

5. REFERENCES

[1] "Developments of the Honeyd Virtual honeypot", http://www.honeyd.org

[2] Lance Spitzner(2002), Honeypots: Tracking Hackers, Addison Wesley.

[3] Robert McGrew, Rayford B. Vaughn(2006), Experiences with Honeypot Systems: Development, Deployment and Analysis, *Proc. IEEEIntl. Conf. System Sciences,* Hawaii

[4] "SPECTER Intrusion Detection System", http://www.specter.com/default50.htm

[5] Vukasin Pejovi, Ivana Kovacevi, Slobodan Bojani, Corado Leita, Jelena Popovi, Octavio Nieto-Taladriz(2007), Migrating a Honeypot to Hardware, *Proc. IEEEIntl. Conf. Emerging Security Information, Systems and Technologies,* Spain, pp.35