ISSN 2395-1621

A Network Defense System for Detecting and Preventing Potential Hacking Attempts

[#1]Hipparkar A.A., [#2]Bhosale S.N., [#3]Shaha N.M.,
[#4]Kokate S.S, [#5]Nanaware P.P.

[1]Assistant Prof., Computer Science, COEP, Phaltan, India
[2,3,4,5]Student, Computer Science, COEP, Phaltan, India

ABSTRACT

A honeypot is closely monitored decoy serving several purposes: it can distract adversaries from more valuable machines on a network, provide early warning about new attack and exploitation trends, or allow in depth examination of adversaries during and after exploitation of a honeypot. Deploying a physical honeypot is often time intensive and expensive as different operating systems require specialization hardware and every honeypot requires its own physical system. This paper presents Honeyd, a framework for virtual honeypot that simulates virtual computer system at the network level. The simulated computer systems appear to run on unallocated network addresses. To device network fingerprinting tools, honeyd simulates the networking stack of different operating systems and can provide arbitrary routing topologies and services for an arbitrary number of virtual systems. This paper discusses Honed's design and shows how the honeyd framework helps in many areas of system security, e.g detecting disabling worms, distracting adversaries, or preventing the spread of spam email.

ARTICLE INFO

Article History

Received: 21st February 2017

Received in revised form :

21st February 2017

Accepted: 23rd February 2017

Published online :

24th February 2017

Keywords: Active defence system, hacking, network security, pre-hacking.

I. INTRODUCTION

Honeypot will act as a middleware for main server and clients. All the communication packets will go through honeypot so that by accessing these packets admin can decide what action to take. Also honeypot will create virtual IPs and assign them to clients few of them are initially marked as vulnerable as per there locations and frequency of use. If honeypot detects request from those IP it will send fake IP of server.

II. LITERATURE SURVEY

Honey'd is a program that allows the user to run virtual hosts on machine on the network, in essence it's basically solely for deploying honeypots. Honey's functions allow it to emulate almost any known operating system at the IP stack level, instead of service level, and multitude of services. It is under license of GNU General Public License, and is freely available to be downloaded and deployed by everyone. The senior staff engineer of Google Inc., Niels Provos, created honey'd and has written highly detailed documentation on how it works. Very surprisingly for how simple it is, honey'd is an extremely powerful program for creating virtual honeypots. Evaluations of honey'd show a 1.1GHz Pentium III processor sustaining over 2,000 TCP transactions per second with a total bandwidth usage of 30 Bits/s. (Provos, 2004) Of course, the reason the program is so lightweight is because it is low-interaction. Instead of simulating every aspect of the OS, honeyed simply copies it at the network stack.

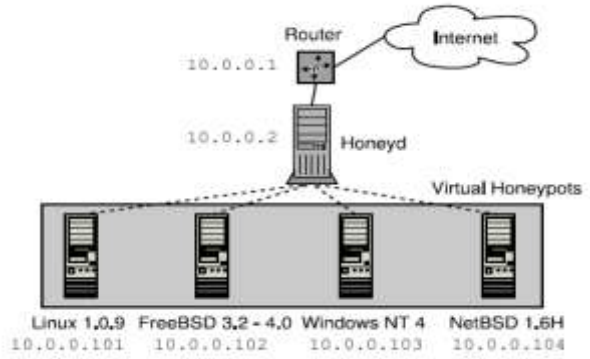
III. PROBLEM DEFINITION

Problem Statement – In network attacker want the address of web server so they can execute their attacks or scripts on the server and access the data of that website. For this problem we need an interpreter or middleware to monitor such activities or detect vulnerable IPs which we propose in our honeypot

framework.

A. Description of problem

Java base honeypot monitoring client and intrusion detection. Honeypot is intermediate between virtual hosts and web server, which captures the information in packet using local area connection. Honeypot filters IP and packet. It is checked by administrator can guess attacker and honeypot detect to attacker and it gives server fake IP.



B. Packet capturing and filtering Algorithm

Algorithm -

JPCAP -

JPCAP can be used to develop many kinds of network applications, including –

- JPCAP uses the LIBCAP & WINPCAP for libraries.
- network and protocol analyzers
- network monitors
- traffic loggers
- traffic generators
- user-level bridges and routers
- network intrusion detection systems(NIDS)
- network scanners & security tools

1. Install JPCAP and WINPCAP

2. Obtain list of network interfaces

To capture packets from a network, obtain list of network interfaces.
JpcapCaptor.getDeviceList(): It returns array of Network interface objects.

3. Open a network interface

Choose which interface to capture packets from, open the interface by using jpcapCaptor.openDevice() method

TABLE. PIECE OF CODE ILLUSTRATES HOW TO OPEN NETWORK INTERFACE

Purpose	**Name**	
Interface	NetworkInterface	1
Max number of bytes to capture at once	int snaplen	2
True if we want to open interface in promiscuous mode, and otherwise false. In promiscuous mode, you can capture packet from the wire In non-promiscuous mode, you can only capture packets send and received by your host.	boolean promics	3
Set a capture timeout value in milliseconds	int to_ms	4

4. Capture packets from the network interface

There are two major approaches to capture packets using JpcapCaptor.getPacket() method.

getPacket() method simply returns a captured packet.

5. set capturing filter

In Jpcap, you can set a filter so that Jpcap doesn't capture unwanted packets. For example, if you only want to capture TCP/IPv4 packets, you can set a filter as following:

The filter expression "ip and tcp" means to "keep only the packets that are both IPV4 and TCP and deliver them to the application".

By properly setting a filter, you can reduce the number of packets to examine, and thus can improve the performance of your application.

IV. CONCLUSION

Honeypot system is growing technique in network security. We are protecting sites from hackers or intruders. In this we have created a virtual honeypot server which prevent the system from attackers or hackers. We use routing protocol.The term virtual used because all different operating systems have the appearance to be running as an independent computer.

ACKNOWLEDGEMENT

Authors would like to express gratitude to **Prof. R. P. Bagawade**, Head of Computer Science and Engineering department for his guidance and support in this work. The authors are also thankful to principal, PES's College of Engineering, Phaltan for being a

constant source of inspiration.

REFERENCES

[1] Alsunbul, S.; Le, P.; Tan, J., A Defense Security Approach for Infrastructures against Hacking, Trust, Security and Privacy in Computing and Communications (TrustCom), 2013 12th IEEE International Conference on , vol., no., pp.1600,1606, 16-18 July 2013.

[2] S. Mcclure, J. Scambray, and G. Kurtz, Hacking Exposed: Hacking Exposed: Network Security Secrets and Solutions, Seventh Edition: McGraw-Hill, Inc.2012.

[3] Vukalovic, J,; Delija, D., " Advanced Persistent Threats - Detection and Defence", Information and Communication Technology, Electronics and Microelectronics (MICRO), 2015 38th International Convention on. vol., pp.1324, 1330, 25-29 May 2015.

[4] Saadaoui, A.; Ben Souayeh, N.B.Y.; Bouhoula, A., "Formal approach for managing rewall misconfigurations," Research Challenges in Information Science (RCIS), 2014 IEEE Eighth International Conference on ,vol., no., pp.1,10, 28-30 May 2014.

[5] Appelt, D.; Nguyen, C.D.; Briand, L., "Behind an Application Firewall, Are We Safe from SQL Injection Attacks?," Software Testing, Verification and Validation (ICST), 2015 IEEE 8th International Conference on ,vol., no., pp.1,10, 13-17 April 2015.

[6] Makiou, A.; Begriche, Y.; Serhrouchni, A., "Improving Web Application Firewalls to detect advanced SQL injection attacks," information Assurance and Security (IAS), 2014 10th International Conference on ,vol., no., pp.35,40, 28-30 Nov. 2014.

[7] Pevny, T.; Komon, M.; Rehaky, M., "Attacking the IDS learning processes," Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on ,Vol., no., pp.8687,8691, 26-31 May 2013.

[8] M. Howard and D. LeBlanc. Writing Secure Code.Microsoft Press, USA, 2nd edition, 2003.

