Leveraging On Honeypot Technology To Improve Network Security
Ayeni, O. A, Alese, B. K, Dada, O. M, Aliyu, E. O

Abstract

In this paper, a new framework is proposed for the design and implementation of decoy honeypot. In this research work, emphasis is laid on ways to improve network security by deploying honeypot technology. The design and configuration of a honeypot was implemented using a virtual machine (VM) ware workstation to detect attack or malicious traffic on a network. VM ware enables the creation, configuration, distribution, support and manages virtual machine similar to the one used on real computer system. Intrusion detection system (IDS), Entropy based detection scheme and Virtual machine (VM) ware work station were used to capture and analyse traffic over the network. The result shows that the deployment of an honeypot successfully fools an attacker to believe he is attacking a real system. Also, it shows that Honeypot can be deployed on a network to help in enhancing system security.

Keyword: honeypot, traffic, network, detection & entropy.

INTRODUCTION.

It is increasingly becoming difficult to secure computer networks due to largely increase in the activities of e-commerce over the internet. In recent times, a lot of losses have been recorded in term of cost and integrity of confidential data has been compromised due to the activities of hackers over the internet.

Today, information is a vital element in every aspect of life. Up-to-date and correct information are the key to any successful businesses, academia, government, personal finances or leisure activities. While this has been true for hundreds of years, it has never been as true as in the last half of the 20th century with the invention of the modern digital computer. Security is one of the hottest issues in network today. Worries about security have soared because of the increasing magnitude of electronic commerce occurring over the Internet and the swiftly evolving business trend towards telecommuting. Therefore, more sensitive and critical information is crossing the world than ever before. (Nor et al, 2006).[27]. The expansion of the World Wide Web has given unlimited access to attackers to prey on ignorant administrator who lacks basic knowledge of network security. Vulnerabilities in common security components such as firewalls, security patches, access control and encryption are inevitable, so hackers take advantage of these loopholes to break into computer networks. This paper presents the result of using a honeypot to limit the activities of hackers/attackers over computer networks. All traffic from and to a honeypot is considered to be unauthorised activity. Compromised honeypots are not threats to the security of the network as long as it is not high interaction honeypot but rather it aids us by collecting the data generated. All data collected by a honeypot is consequently interesting data. Data gathered by a honeypot is valuable and can lead to a better understanding and awareness which in turn can assist administrator in increasing overall network security.

Security of computing machines and networks are increasing in importance as more and more business is conducted via these systems. Despite decades of research and experience, we are still unable to make secure computer systems or even manufacture ways to measure a computer system’s level of security. The automation of exploit scripts and massive global scanning for vulnerabilities enable adversaries to compromise computer systems shortly after vulnerabilities become known. One way we can strengthen our defenses against these scans and exploit scripts is to learn from our adversaries by deploying and monitoring computer systems on a network that we expect to be broken into. These machines or systems we manufacture to be broken into are called Honeypots. When studying our adversaries we need to monitor and log every connection attempt to these machines and the known vulnerabilities present in our deployments.

1. Honeypot

The basic idea of a honeypot is quite old and has been in used already for quite a long time. Prior to honeypots, there was the seminal narrative by
Clifford Stoll of monitoring and tracking an intruder (Stoll, 1998).[32]. Stoll 1998 described how he created a complete but non existent government project with realistic but false files which attackers spent an extended period of time downloading and analyzing, thus providing an opportunity to monitor and trace the attackers. The original Honeypot computer systems are documented in the two proceedings that are presented by Bellovin and Cheswick (Bellovin, 1992 and Cheswick, 1992) [7]. Although, the word “Honeypot” is a new phrase but the technology is not new and is getting more and more crucial. Possible definitions of what a honeypot is:

Spitzner, Lance (2003). [30] defines the term “Honeypot” as follows: A honeypot is a resource whose value is being attacked or compromised. This means, that a honeypot is expected to get probed, attacked and potentially exploited. Honeypots do not fix anything. They provide us with additional, valuable information.

2 Intrusion detection system

Intrusion detection, according to Kuwatly et al (2003) [17], is the process of monitoring computers or networks for unauthorized entrance or activity. Intrusion Detection System (IDS) can be used to monitor network traffic, thereby detecting if a system is being targeted by a network attack. There are two basic types of intrusion detection: host-based (HIDS) and network based (NIDS). Each has a distinct approach to monitoring and securing data, and each has distinct advantages and disadvantages. Host-based IDSs examine data held on individual computers that serve as hosts; they are highly effective for detecting insider abuses. Examples of HIDS implementations include Windows NT/2000 Security Event Logs, and UNIX Syslog. On the other hand, Network-based intrusion detection systems analyze data packets that travel over the actual network. These packets are examined and sometimes compared with empirical data to verify whether they are of malicious or benign nature. An example of NIDS is Snort, which is an open source software tool that can capture real-time network traffic. It can be configured to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, and OS fingerprinting attempts.

A simple virtual network of three systems will be design to implement the concept of the system. The first system on the network served as a gateway (point of presence) to the other two systems, it is the responsibility of this system to redirect traffic flow to either the virtual server or the honeypot. This system will be connected to the internet to allow interaction with the network. The second system will serve as a virtual server with five hundred ports opened to imitate application services available and running on the system. The purpose of opening several port is to make clients and attackers believe there are real and important services running on the system. The third system will be the honeypot system which will be used to performs analysis on the attack traffic flow arriving into the network.

The configuration of the POP server include: 2000MB of memory, 15GB hard disc (SCSI), CD-ROM (IDE), single processor (Pentium(R) ), Windows Server 2003 Enterprise edition. A software called snort will be installed to extract packet data information from traffic flow and perform entropy test to determine if a traffic flow is an attack or legitimate.

The requirement for the design of the medium-level dynamic honeypot system include: An operating system such as Windows 2003 Professional, with a 1GHz processor, 512 Mb of RAM, with a 10/100 network card already and a CDROM or DVD/RW drive. Windows 2003 Professional was the best choice to since it can be secured the most from the operating systems, other operating system that can be use include: Windows XP, Windows 2000 Server and Windows 2003 Professional. A program called Snort will be installed in the system. This program is an open source network intrusion prevention and detection system utilizing a rule-driven language, which combines the benefits of signature, protocol and anomaly based inspection methods. Snort is the most widely deployed intrusion detection and prevention technology worldwide and has become the de facto standard for the industry. (http://www.snort.org/) Snort is a free program which is extremely powerful in what it does. This is part of an intrusion detection system. Honeypot works by opening over 1000 user datagram protocol (udp) and transmission control protocol (tcp) listening sockets on the computer and these sockets are designed to mimic vulnerable services. When an attacker connects to these services they are fooled into thinking they are attacking a real server. The honeypot safely captures all communications with the attacker and logs these results for future analysis.

3. PROPOSED SYSTEM

The proposed framework provides for proactive mitigation against the effects of DoS attack. A system designed to redirect network traffic flow is positioned at the network gateway as point of presence (POP). All the traffic flows arriving at the Point of Presence (POP) of a destination network server to be protected from DoS attack are tagged as either legitimate or attack. Whenever a packet belonging to suspicious flow arrives at the POP, instead of sending that packet to the active FTP server or dropping it, it is redirected to honeypot...
This provides a proactive approach to mitigation against the attack because the FTP server is isolated from attack traffic and bandwidth of the links with FTP server will not be exhausted by the voluminous attack traffic.

![Proposed system diagram](image)

**Fig.1** Proposed system.

The topology is similar to the one used to depict a typical client-server scenario in the Internet. The clients (attack and legitimate) send their FTP requests to the server. The arrows in the figure indicate the presence of variable rate attacks coming from client domain.

### 3.1 Entropy based detection

The system detect and characterize attacks treats DoS anomalies as events that disturb the distribution of traffic features. For example, a DoS attack, regardless of its volume, will cause the distribution of destination address to be concentrated on the victim address or server. As proposed by Sardana et al (2008), entropy scheme will capture the degree of dispersal or concentration of a distribution flow. The sample entropy $H(X)$ is

$$H(X) = - \sum_{i=1}^{N} P_i \times \log_2(P_i)$$

where $P_i = N_i/S$. $N$ is a set of positive integer that represents total number of system (server) on the network, $P_i$ represents a flow of traffic at i. The value of sample entropy lies in the range range $0$ through $\log_2 N$. The metric takes on the value $0$ when the distribution is maximally concentrated, i.e., all observations are the same. Sample entropy takes on the value $\log_2 N$ when the distribution is maximally dispersed. Let $NS$ and $NH$ represent the number of servers and honeypots respectively. A variable $lengvec$ is defined such that $lengvec = NS + NH$. An array vector $[i]$ of size $lengvec$ is also defined whose elements are in the form of ordered pair set of destination IP address and port number of the honeypot or the server i.e. vector$[i] = \{dest IP, port\}$. Let $S$ represent the set of indexes of the vector$[i]$ array. Further, two arrays $subvecNS [i]$ and $subvecNH [i]$ is also define whose elements are indices of the array vector$[i]$ that correspond to destination IP address and port number of servers and honeypots respectively such that the following holds true:

$lengvec = (Length(subvecNS) + Length(subvecNH)) \text{AND} (subvecNS \cap subvecNH)$

**Algorithm**

The redirection algorithm performs the per-flow treatment of each flow in the Flow List (FL) in a time window at POP.

- FDA – Flow Destination Address
- FSA – Flow Source Address
- NDA – Network Destination Address
- PDA – Packet Destination Address

The pseudo code is as follows:

**Honeypot Controller PerFlow (FL)**

**For a flow in FL**

If (flow Tag = attack)

Parse the primary packet and search source and destination address (FDA and FSA)

PDA = FDA

NDA = PDA

A: If (NDA = Destination address of honeypot)

Forward the packet to NDA

Else

Replace NDA by destination address of honeypot

Forward the packet to NDA

If (More Fragment = 0)

Goto S

Else

Parse next header of the flow for PDA

NDA = PDA

If (Tag = attack)

Goto A

Else

Goto B

Else

Parse the primary packet and search source and destination address (FDA and FSA)

PDA = FDA

NDA = PDA

B: If (NDA = Destination address of active FTP server)

Forward the packet to NDA
Else
Replace NDA by destination address of server
Forward the packet to NDA
If (More Fragment = 0)
Goto S
Else
parse next header of the flow for PDA
NDA = PDA
If (Tag = attack)
Goto A
Else
Goto B
S: Stop

POP Server Sniffer and Detection Mode
The POP server which direct network traffic to the other two systems is configured and executed in sniffer mode to extract network traffic details from incoming network connection. If the packet extracted from the connection is malicious, it is directed to the honeypot otherwise it directs the packet to the virtual server.

Protocol: UDP
Source: 192.168.100.161 (192.168.100.161)
Destination: 192.168.0.132 (192.168.0.132)
User Datagram Protocol, Src Port: 54296 (54296), Data (991 bytes)

The data below shows the result from the network packet from honeypot system

08/21-02:55:40.702060 [**] [1:1000002:1]
extracting packet data from network connection attk[**] [Priority: 0] {TCP} 192.168.0.129:1048 -> 192.168.0.132:80
extracting packet data from network connection attk[**] [Priority: 0] {TCP} 192.168.0.129:1048 -> 192.168.0.132:80
08/21-02:55:41.626241 [**] [1:1000002:1]
extracting packet data from network connection attk[**] [Priority: 0] {TCP} 192.168.0.129:1048 -> 192.168.0.132:80
08/21-02:55:42.141453 [**] [1:1000002:1]
extracting packet data from network connection attk[**] [Priority: 0] {TCP} 192.168.0.129:1048 -> 192.168.0.132:80
08/21-02:55:42.141453 [**] [1:1000002:1]
extracting packet data from network connection attk[**] [Priority: 0] {TCP} 192.168.0.129:1048 -> 192.168.0.132:80

The data below shows the result from the network packet from honeypot system

08/21-02:55:40.702060 [**] [1:1000002:1]
extracting packet data from network connection attk[**] [Priority: 0] {TCP} 192.168.0.129:1048 -> 192.168.0.132:80
extracting packet data from network connection attk[**] [Priority: 0] {TCP} 192.168.0.129:1048 -> 192.168.0.132:80
08/21-02:55:41.626241 [**] [1:1000002:1]
extracting packet data from network connection attk[**] [Priority: 0] {TCP} 192.168.0.129:1048 -> 192.168.0.132:80
08/21-02:55:42.141453 [**] [1:1000002:1]
extracting packet data from network connection attk[**] [Priority: 0] {TCP} 192.168.0.129:1048 -> 192.168.0.132:80
08/21-02:55:42.141453 [**] [1:1000002:1]
extracting packet data from network connection attk[**] [Priority: 0] {TCP} 192.168.0.129:1048 -> 192.168.0.132:80

Result
The following result was obtained in a log file after about two hour of connecting the virtual network to the internet.

The data below shows the sample result in log file from the network packet at POP server:

00000000 4d 45 53 47 00 00 00 64 4e 41 4d 45 00 00 10 [MESG...dNAME....]

Fig.2 Network packet at PoP server
Mining. Network connection summary from logfile

<table>
<thead>
<tr>
<th>Time</th>
<th>Source IP</th>
<th>Destination IP</th>
<th>Protocol</th>
<th>Category</th>
<th>Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>08/21-02:55:42.410615</td>
<td>192.168.0.132</td>
<td>209.85.143.99</td>
<td>TCP</td>
<td>atk</td>
<td>80</td>
</tr>
<tr>
<td>08/21-02:55:42.438233</td>
<td>192.168.0.130</td>
<td>209.85.143.99</td>
<td>TCP</td>
<td>atk</td>
<td>80</td>
</tr>
<tr>
<td>08/21-02:55:42.620976</td>
<td>192.168.0.130</td>
<td>209.85.143.99</td>
<td>TCP</td>
<td>atk</td>
<td>80</td>
</tr>
<tr>
<td>08/21-04:55:40.702060</td>
<td>192.168.0.130</td>
<td>209.85.143.99</td>
<td>TCP</td>
<td>atk</td>
<td>80</td>
</tr>
<tr>
<td>08/21-04:55:41.830478</td>
<td>192.168.0.130</td>
<td>209.85.143.99</td>
<td>TCP</td>
<td>atk</td>
<td>80</td>
</tr>
<tr>
<td>08/21-04:55:41.626241</td>
<td>192.168.0.130</td>
<td>209.85.143.99</td>
<td>TCP</td>
<td>atk</td>
<td>80</td>
</tr>
<tr>
<td>08/21-04:55:42.141453</td>
<td>192.168.0.130</td>
<td>209.85.143.99</td>
<td>TCP</td>
<td>atk</td>
<td>80</td>
</tr>
<tr>
<td>08/21-04:55:42.410615</td>
<td>192.168.0.130</td>
<td>209.85.143.99</td>
<td>TCP</td>
<td>atk</td>
<td>80</td>
</tr>
<tr>
<td>08/21-04:55:42.141453</td>
<td>192.168.0.130</td>
<td>209.85.143.99</td>
<td>TCP</td>
<td>atk</td>
<td>80</td>
</tr>
<tr>
<td>08/21-04:55:42.410615</td>
<td>192.168.0.130</td>
<td>209.85.143.99</td>
<td>TCP</td>
<td>atk</td>
<td>80</td>
</tr>
</tbody>
</table>

**Sample of data obtained from POP logfile**

<table>
<thead>
<tr>
<th>Num</th>
<th>Protocol</th>
<th>Category</th>
<th>Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Tcp</td>
<td>atk</td>
<td>95</td>
</tr>
<tr>
<td>2</td>
<td>Tcp</td>
<td>atk</td>
<td>130</td>
</tr>
<tr>
<td>3</td>
<td>Tcp</td>
<td>atk</td>
<td>497</td>
</tr>
<tr>
<td>4</td>
<td>Tcp</td>
<td>atk</td>
<td>697</td>
</tr>
<tr>
<td>5</td>
<td>Tcp</td>
<td>norm</td>
<td>80</td>
</tr>
<tr>
<td>6</td>
<td>Tcp</td>
<td>atk</td>
<td>497</td>
</tr>
<tr>
<td>7</td>
<td>Tcp</td>
<td>norm</td>
<td>85</td>
</tr>
<tr>
<td>8</td>
<td>Udp</td>
<td>atk</td>
<td>130</td>
</tr>
<tr>
<td>9</td>
<td>Tcp</td>
<td>atk</td>
<td>130</td>
</tr>
<tr>
<td>10</td>
<td>Tcp</td>
<td>atk</td>
<td>100</td>
</tr>
<tr>
<td>11</td>
<td>Tcp</td>
<td>norm</td>
<td>100</td>
</tr>
<tr>
<td>12</td>
<td>Udp</td>
<td>atk</td>
<td>80</td>
</tr>
<tr>
<td>13</td>
<td>Udp</td>
<td>atk</td>
<td>123</td>
</tr>
<tr>
<td>14</td>
<td>Udp</td>
<td>atk</td>
<td>123</td>
</tr>
<tr>
<td>15</td>
<td>Tcp</td>
<td>atk</td>
<td>123</td>
</tr>
<tr>
<td>16</td>
<td>Tcp</td>
<td>atk</td>
<td>258</td>
</tr>
<tr>
<td>17</td>
<td>Tcp</td>
<td>atk</td>
<td>85</td>
</tr>
<tr>
<td>18</td>
<td>Tcp</td>
<td>atk</td>
<td>75</td>
</tr>
</tbody>
</table>

**Network connection summary from logfile**

<table>
<thead>
<tr>
<th>Incoming connection at POP</th>
<th>Honeypot traffic</th>
<th>Virtual server request</th>
</tr>
</thead>
<tbody>
<tr>
<td>323</td>
<td>241</td>
<td>82</td>
</tr>
</tbody>
</table>

### Table 2. Network connection summary from logfile.

#### 4. CONCLUSION

In this research, a medium-interaction honeypot was designed. A simple virtual network of three systems was designed to implement the concept of the system. This system was connected to the internet to allow interaction with the network. The design and configuration of this honeypot was implemented using a virtual machine (VM ware) workstation to detect attack or malicious traffic on a network. The Point of Presence (POP) server serve as a link to the two other systems; honeypot system and the application server with virtual application running in it to give the impression of the presence of useful resources. An intrusion detection software called Snort were installed on each of the component of the honeypot to extract packet data information from traffic flow and perform entropy test on them to determine if a traffic flow is an attack or legitimate. The framework is aware of two internet protocols: TCP and UDP. Packets for other protocols are logged and silently discarded. At the end, malicious traffic were discovered and forwarded to honeypot for further analysis.

### REFERENCES

CERT Statistics http://www.cert.org/starts/cert


[8]. Christian Döring: Improving network security with Honeypots, pages B34-B36, Master's thesis University of Applied Sciences Darmstadt, Department of Informatics, 2005

[9]. Clifford Stoll: The Coocoo’s egg, Pocket Books 1990


[11]. Eric Peter, epeter(at)wustl(dot)edu and Todd Schiller, tschiller(at)acm(dot)org (A project report written under the guidance of Prof Raj Jain). A practical guide to honeypot.


[21]. National Institute of Standards and Technology (NIST)“Guidelines on firewalls and firewall policy” January 2002

[22]. Nor Badrul Anuar, Omar Zakaria, and Chong Wei Yao University of Malaya, Kuala Lumpur MY Honeypot through Web: The emerging of security application integration.


[27]. Saleh Ibrahim Bakr Almotairi. Using honeypots to analyse Anomalous internet activities.


[42]. A virtual Honeypot framework, Niels proved Google, Inc. niel@google.com