Personal Security System

Shaikh Mohd Abdul Hannan[1], Umar Shakeeb[2], Varad Deshpande[3], Sanket Milke[4]

[1] UG Scholar, Dept. Of CSE, MGM JNEC Aurangabad, Maharashtra, India
[2] UG Scholar, Dept. Of CSE, MGM JNEC Aurangabad, Maharashtra, India
[3] UG Scholar, Dept. Of CSE, MGM JNEC Aurangabad, Maharashtra, India
[4] Assistant Professor, Dept. Of CSE, MGM JNEC Aurangabad, Maharashtra, India

ABSTRACT

Nowadays corporate companies' networks can generate false alarms and are a major target of exploits. They have lots of sensitive data that can be misused to leak information critical to the company and its employees. In order to avoid these kinds of attacks, companies use an Intrusion Detection System. An intrusion Detection System (IDS) inspects every packet passing through the network and raises an alarm if there is any attempt to perform malicious activity. IDS ensures a security policy in every single packet passing through the network. Snort is an open-source, lightweight tool that captures every detail of a packet passing through the network and generates alerts if anyone's packet matches the signatures inserted given by the company. The signatures are the rules written so that IDS can know which packets it should generate the alert. In this paper, we have implemented an Intrusion Detection and Prevention System using libpcap, and snort l in order to detect network-based attacks.

1. Introduction

Network security is one of the biggest challenges that companies are facing from time to time. There are many attempts by black hat hackers to break and compromise the security of the Company's network, some of which are even successful. As the use of the internet increasing, these malicious activities are gaining popularity among the black hats.

Every day a large amount of data is being generated and passed on and lots of this data holds sensitive information about the company and its employees. Thus, securing a network is one of the most important tasks for a company to survive. To make this easier and more efficient we use Intrusion Detection System, it helps to collect information about any malicious packet that passes across a company's network.

1.1 Intrusion Detection System

An intrusion detection system (ID) is a type of security system for computers and computer networks. Intrusion Detection basically helps in detecting outer and inner attacks performed by either users or hackers. An ID system collects information from various sources and analyzes information from various areas within a computer or a network to identify possible security breaches, which include both intrusions (attacks from outside the organization) and misuse (attacks from within the organization). ID uses vulnerability assessment (sometimes referred to as scanning), which is a technology developed to assess the security of a computer system or network

1.2 Advantages of IDS

- Track any changes in the behavior of the network.
- Inspects system activity
- Can differentiate between normal and abnormal activities in the network
- Automated

1.2 Disadvantages of IDS

- Sometimes gives false alarms i.e., the packet wasn't malicious but IDS might still generate an alert.
- Time-consuming
- Is not 100% safe from attacks

2. Tools used in IDS

There are various tools to implement an Intrusion detection system. Some of the most widely used tools are

- SNORT
- Security Onion
- WEKA
- OSSEC

3. Personal System System

PSS is a light-weight intrusion detection tool that logs the packets coming through the network and analyzes the packets. Snort checks the packets coming against the rules written by the user and generates alerts if there are any matches found. The rules are written by the user in a text file that is linked with a project.conf file where all the configurations are mentioned. There are a few commands which are used to get project running so that it can analyze network behavior.

3.1 Advantages of Personal Security System over other tools.

1. Scalability: PSS can be successfully deployed on any network environment.

2. Flexibility and Usability: Snort can run on various operating systems including Linux, Windows, and Mac OS X.

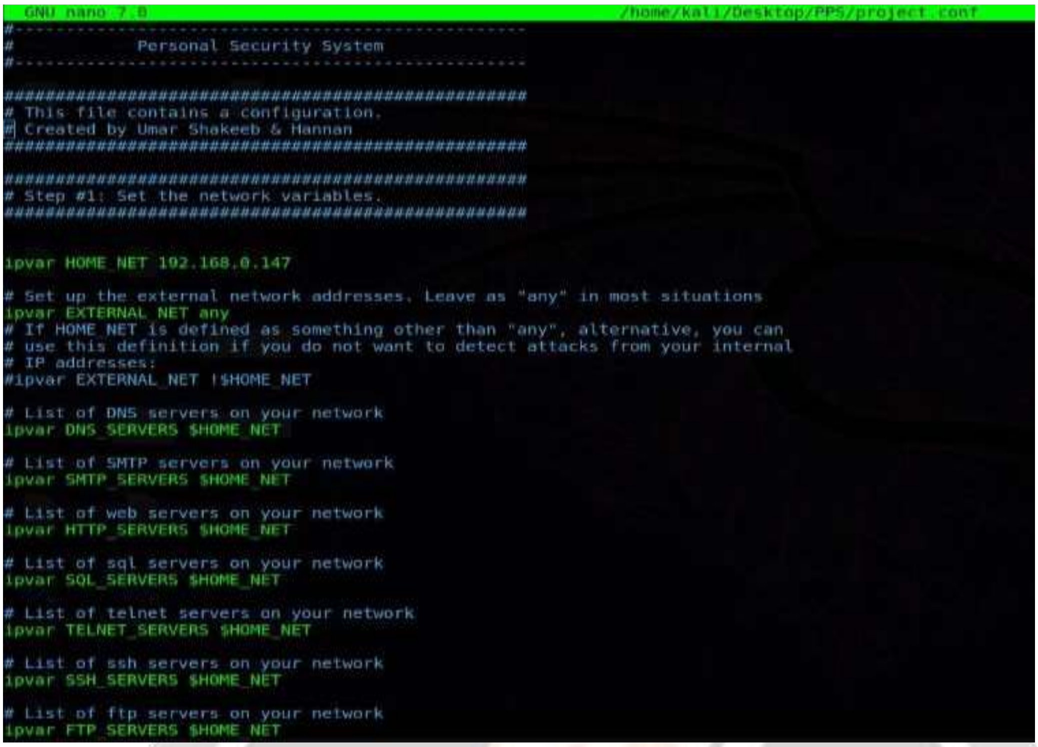
3. Live and Real-Time: PSS can deliver real-time network traffic event information.

4. Flexibility in Deployment: There are thousands of ways that PSS can be deployed and a myriad of databases,

logging systems, and tools with which it can work.

5. Speed in Detecting and Responding to Security

Threats: Used in conjunction with a firewall and other layers of security infrastructure, PSS help organizations detect and respond to system crackers, worms, network vulnerabilities, security threats, and policy abusers that aim to take down network and computer systems.



```
  GNU nano 7.0                                    /home/kali/Desktop/PPS/project.conf
#---------------------------------------------------
#             Personal Security System
#---------------------------------------------------

###################################################
# This file contains a configuration.
# Created by Umar Shakeeb & Hannan
###################################################

###################################################
# Step #1: Set the network variables.
###################################################


ipvar HOME_NET 192.168.0.147

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
# If HOME_NET is defined as something other than "any", alternative, you can
# use this definition if you do not want to detect attacks from your internal
# IP addresses:
#ipvar EXTERNAL_NET !$HOME_NET

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET

# List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET

# List of ssh servers on your network
ipvar SSH_SERVERS $HOME_NET

# List of ftp servers on your network
ipvar FTP_SERVERS $HOME_NET
```

Configuration File

4. Personal Security System using project.conf file

PSS uses a configuration file at start-up time. A sample configuration file snort. You use the - c command line switch to specify the name of the configuration file. The following command uses /opt/snort/PSS.conf as the configuration file.

We can also save the configuration file in our home directory as PSS, but the most commonly used method is specifying it on the command line. There are other advantages to using the configuration file name as a command line argument to a project. It is possible to invoke multiple Snort instances on different network interfaces with a different configuration.

```
$ sudo -A console -q -i wlan0 -c /etc/snort/snort.conf
```

This command should be run in our terminal to run Personal Security System using our project configuration file. It can be modified according to user suitability. Snort library has various modes; a few of them are listed here

Description of the command:

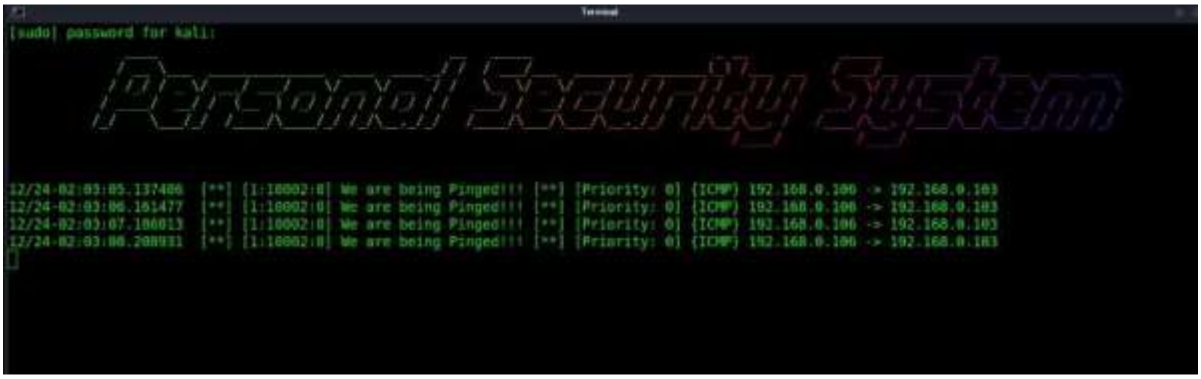
-c: specifies the config file

-i: specifies the interface mode, if a loopback address is running then “wlan0” will be written, for Ethernet “eth0” or “eth1” will be written.

-A: It will print the output to the console

Once we run this command, then type $ ping 127.0.0.1

We should see that the project. Logs this packet and displays it on the terminal. Here is the image of the terminal logging the ping packets.



```
[sudo] password for kali:

12/24-02:03:05.137486  [**] [1:10002:0] We are being Pinged!!! [**] [Priority: 0] {ICMP} 192.168.0.106 -> 192.168.0.103
12/24-02:03:06.161477  [**] [1:10002:0] We are being Pinged!!! [**] [Priority: 0] {ICMP} 192.168.0.106 -> 192.168.0.103
12/24-02:03:07.186013  [**] [1:10002:0] We are being Pinged!!! [**] [Priority: 0] {ICMP} 192.168.0.106 -> 192.168.0.103
12/24-02:03:08.208931  [**] [1:10002:0] We are being Pinged!!! [**] [Priority: 0] {ICMP} 192.168.0.106 -> 192.168.0.103
```

Monitoring Screen

4. Writing rules: -

Rules are written by the user, It generates an alert if there if finds any match with the rules that the user defined in the rules file. Here is an example of how to write rules.

1. alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg:"We are being Pinged!!!";icode:0; itype:8; sid:10002;)

   for ICMP ping.
2. alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS 80 (msg:"Triggered /etc/passwd"; flow:to_server,established; content:"/etc/passwd"; nocase; sid:1122;)

   When someone try OS command Injection
3. alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"XSS attack attempted (cross site scripting attempt)"; flow:to_server,established; content:"SCRIPT"; nocase; sid:1497;)

   When XXS (Cross Site Scripting) attack is perform

4. alert tcp $EXTERNAL_NET any -> $HTTP_SERVERS $HTTP_PORTS (msg:"COMMUNITY SQL-INJECTION BXCP Sql Injection attempt"; flow:to_server,established; uricontent:"/index.php"; nocase; uricontent:"where="; nocase; uricontent:"union"; nocase; uricontent:"select"; nocase; sid:100000690; rev:2;)

   When SQL injection attack is performed

5. alert tcp any any -> $HOME_NET 80 (flags:S; msg:"Possible Dos Attack Type:SYN FLOOD"; flow:stateless; sid:3; detection_filter:track by_dst, count 20, seconds 10;)

   When DOS (Denial of Service) attack is performed

6. alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg:"Incoming FTP connection!!!"; flags:S; sid:10000;)

5. Result

The result of our project will be the display of all packets which match the snort defined by the administrator. The information which gets on Monitor Screen is Source IP, Destination IP, Alert generated, and Date and Time when the packet was received. In this case, we have used a single system for testing purposes, therefore, the source and destination IP are my loopback address, however, when run on a server it will give the Source and Destination IP of the systems generating and receiving the packets. In this case, we have used a single system for testing purposes therefore the source and destination IP are my loopback address, however, when run on a server it will give the Source and Destination IP of the systems generating and receiving the packets

6.CONCLUSIONS

The goal of this paper was to design and evaluate a system that would passively monitor the wireless network traffic of a small home network. Such a system, in case of attack detection, attempts to disrupt detected attacks using the packet injection method. During experiments was an attempt to carry out DoS by ICMP flood on the access point. This type of attack was chosen because of its simple detection and execution. The output of Snort statistics was shown decrease of attacker's traffic by 95% with proposed IDPS system deployed – when compared to the amount of attacker's traffic without deployment of IDPS. From the results we conclude that the deauthentication of the attacker successfully disassociates the attacker from the access point, and thus prohibits the attack. Since the initiation of the attack the system was able to react in 0.2 seconds, as is concluded from graphs generated by Wireshark statistics. Aireplay-ng standard output was used to calculate statistics of deauthentication. From these statistics is assumed that the deauthentication of the attacker was in all cases successful, nevertheless, the percentage of received deauthentication acknowledgments from the attacker was only 42% when compared to the 100% of received acknowledgments from the access point.

7.REFERENCES

[1] Karen, Scarfone & Peter Mell, (2007) "Guide To Intrusion Detection And Prevention Systems (IDPS)". Washington, D.C.: National Institute of Standards and Technology, Special Publication 80094, 128 p.

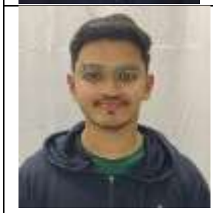
[2] Michael Rash, (2007) "Linux Firewalls - Attack Detection And Response With Iptables", Psad And Fwsnort. San Francisco: No Starch Press, 388 p.

[3] Allen, Lee (2012) "Advanced Penetration Testing for Highly--Secured Environments: The Ultimate Security Guide". Birmingham: Packt Publishing Ltd., 414p.

[4] "Linux Wireless - Hostapd Linux Documentation Page". [online]. [cit. 14. April. 2014]. Available online: <http://wireless.kernel.org/en/users/Documentation/hostapd>.

[5] KAZIENKO, Przemyslaw; DOROSZ, Piotr. Intrusion detection systems (IDS) Part 2-Classification; methods; techniques. WindowsSecurity. com, 2004. [10] CARL, Glenn, et al. Denial-of-service attack-detection techniques. Internet Computing, IEEE, 2006, 10.1: 82-89.

8.Authors

	Sanket Milke, Assistant Professor since 2014, Department of Computer Science & Engineering in MGM Jawaharlal Nehru Engineering College, Aurangabad, Maharashtra. 8 years of Teaching Experience and has Published two International Journal Paper with 1 year of working Experience in Service Based Company. His Primary interest are Cloud Computing, Computer Architecture, Computer Networking, Theory of Computation, Programming in Python & R.
	Umar Shakeeb, Ethical Hacker & Cyber Security Analyst. Pursuing Undergraduate Degree in Computer Science & Engineering from MGM Jawaharlal Nehru Engineering College, Aurangabad, Maharashtra. His primary interests are software development, web development, Python, networking, network Security, Cyber Security, Cloud Computing & Big Data
	Varad Deshpande, Programmer Analyst. Pursuing Undergraduate Degree in Computer Science & Engineering from MGM Jawaharlal Nehru Engineering College, Aurangabad, Maharashtra. His primary interests are software development, web development, programming in PHP and Python, Java, C, C++, Vb.net Working on Full stack Projects.
	Shaikh Mohd Abdul Hannan, Ethical Hacker & Cyber Security Analyst. Pursuing Undergraduate Degree in Computer Science & Engineering from MGM Jawaharlal Nehru Engineering College, Aurangabad, Maharashtra, His primary interests are software development, web development, programming in PHP and Python, networking, network security, System Administration, Cloud Computing & Big Data