Stealth Port Scan Detection Using Snort

Muthyam G Reddy
1228244
Instructor - Dr. Tokunbo Makanju
Information Network and Computer Security
Newyork Institute of Technology
Vancouver, British Columbia
Email: mgreddy@nyit.edu

Hari Priya Aalla
1231151
Instructor - Dr. Tokunbo Makanju
Information Network and Computer Security
Newyork Institute of Technology
Vancouver, British Columbia
Email: haalla@nyit.edu

Abstract—**Port scanning is a reconnaissance phase in the planning of cyber-attacks and many researchers have implemented different techniques to secure the network from port scans. Before carrying out an attack, an attacker often tries to scan a network, to identify its properties or to compromise it. The proposed way to stop these kinds of scans is through detecting the scans and alerting users. Intrusion detection systems can recognize the pattern using source address and the target address to detect an attack. But stealth scan attacks like idle port scan are not easy to find because they use spoofed IP address for port scanning. And since the activities like packet logging are alike TCP connect (), the false positive rate for the detection may increase severely. In this paper, we propose an IP Identification based detection plug-in to detect idle port scan attack using FIN and RST flags with the IP ID number of the packet.**

I. Introduction

Snort is an open source network-based intrusion detection/prevention system that has the ability to perform protocol analysis, content searching and matching [1].

Port scanning is a method for determining open ports in a network, it is analogous to knocking on doors to see if someone is home, and to check whether the port is open and listening. The kind of response received indicates whether the port can be probed for further security weaknesses and also helps us to find out which firewalls are present in between the sender and the target [2]. Network scanning attacks are observed around 50 % against cyber systems [3].This is not an offence until the motive of port scan is not intrusive, because this is also used by security experts to perform penetration testing [4].

In this paper, we are using snort to detect all types of stealth port scan attacks, by writing snort rules. For detecting Idle stealth port scan attack by snort, we have to create a detection plugin which should be smart enough to find out the Idle scan packet array. The snort modules are encoded in C. Once after integrating the detection plugin inside snort, snort will be fully able to detect and alert all types of stealth port scan attacks.

In the following sections, we have discussed the performance of idle port scan. For detection of the idle port scan, there are two phases: packet capturing and preprocessing phase, and analysis and detection phase. We have implemented these phases using the pseudo code. Next step is creating snort rules and observing alerts.

The remaining sections are structured as fallows:Information about port scans in section 2, in section 2 and section we talked about soutions and methodology. in section 5 Results and last conclusion.

II. Related Work

By analyzing the different tools for port-scanning, it can be concluded that with the priority option in snort, the attacks can be detected, and this priority level can be used to enhance the prevention mechanism. Detection and prevention of port-scan attack can neutralize the possibility of future attacks on the cloud environment [5].

A Zero-day attack is a critical network attack. The zero-day attack period (ZDAP) is the period from the release of malware/exploit until a patch becomes available. IDS/IPS cannot effectively block zero-day attacks because they use pattern-based signatures in general. Prior to the actual attack, hackers scan networks to identify hosts with vulnerable ports. If this port scanning can be detected early, zero-day attacks will become detectable. In [6] researchers proposed a Prophetic Defender (PD). PD architecture makes use of a honeypot-based pseudo server deployed to detect malicious port scans.

Although firewalls and router-based packet filtering are essential elements of an overall network security topology, they are not enough on their own. So, to brace the network from unauthorized access the idea of Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) is attracting security experts. Being an open source IDS, Snort can be easily configured and deployed in any environment [7].

As networks become faster, there is a need for intrusion detection systems that can gather all traffic in a network. Researchers Afshin and Leila proposed a collaborative architecture for snort to improve the power of snort in high-speed networks [8]. They utilized a preprocessor system in entrance of network and placed snort in a host and also placed a knowledge-base in preprocessor to send traffic property to specific snort. This increases the snorts ability to work in high-speed networks and makes it able to detect identical attacks quickly without computing.

As one of the powerful and lightweight Network Intrusion Detection System, Snort has good expansibility and transplantability and can be used in various environments. However, Snort has clear deficiency on ARP (Address Resolution Protocol) spoofing detection, and its own ARP spoofingdefense is powerless. To realize the detection and prevention of the ARP spoofing, researchers in the paper [9] expanded the snort preprocessor plug-ins by adding an ARP detection module. Results show this way Snort sniffer has immunity and make locate the attacker more timely and accurately.

In IEEE paper, [10] authors have worked on creating the detection plugin in snort for detecting Idle stealth port scan. We are implementing the proposed model and adding three more stealth scan attacks namely FIN, XMAS and NULL in the rule set.

A. TCP Flags

There are six flags in a TCP packet SYN, RST, FIN, PSH, ACK and URG

a) SYN(synchronization flag): is used initially to sent establishing the three-way handshake between two hosts.

b) RST(reset flag): is used when a segment arrives that is not intended for the current connection.

c) FIN: is used to close the connection.

d) PSH(push flag): tells the receiving end of the TCP connection to push all the buffered data to the receiving application.

e) ACK(acknowledgement flag): is used to acknowledge the successful receipt of packets

f) URG(urgent flag): is used to identify incoming data as urgent.

B. Port Scan Attacks

Basically, port scans are categorized into two types [12]:

a) Stealth Port scan: Any kind of port scan that bypasses a firewall or router and behaves like a casual network is considered as stealth port scanning. FIN scan, XMAS scan and NULL scans are common types of stealth port scans [3].

b) Non-Stealth Port scan: This is a process to identify open ports in a host. These types of scans are easily detected by routers and firewalls and complete the three-way handshake process of a port scan [11].

The various kinds of stealth port scans are as follows:

• FIN scan: In FIN scan, the packets are sent to the target port with FIN flag set. If the target port is open, we do not get any response. If the port is closed, we get a RST packet.

• XMAS scan: In XMAS scan we send packets with FIN, URG and PSH flags set. If the target port is closed, we get the RST packet in response. But, if the port is open, we do not get any response.

• Null Scan: It is also called pre-attack probe, which is a series of TCP packets contain a sequence number of 0 and no flags set. For which the response will be none because none of the flags has been set.

• Idle Stealth Port Scanning: This scan uses the spoofed IP address and sends the SYN packets to the target to find out the details about the target, by observing the IP ID (Internet Protocol Identification number).

The mechanism of Idle port scan is as follows:

In an Idle scan, an attacker can attack a system without revealing its original IP ID. While performing Idle scan, the attacker can attack a system by sending requests from a zombie machine without revealing its true IP address. Steps for performing Idle port scanning for an open port are as follows:

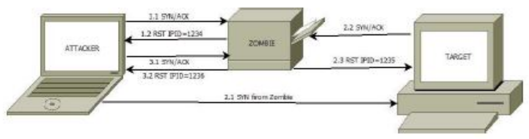


Fig. 1. Idle Stealth Port Scan for Open Port

Step 1. To record IP ID of the Zombie, probe the Zombie machine

The IP ID field is a serial number of the IPv4 packet. To obtain the IP ID of the zombie, the attacker is sending an SYN/ACK packet to Zombie. As a response to attacker, Zombie sends back RST with IP ID.

Step 2. Attacker forges an SYN packet from the zombie. The attacker sends SYN to target that appears to come from the zombie. The Zombie which is not expecting an SYN responds to target with an RST with IP ID.

Step 3. To record the IP ID of the Zombie, probe the Zombie machine again.

Attacker probes the zombie again, by sending SYN/ACK, the zombie responds with RST and IP ID. Since step 1, the IP ID has been increased by 2, indicating the port is open.

If the target port is closed, in step 2 target sends RST packet to the zombie, for which zombie will not respond. When the attacker probes zombie again, it will send an RST packet with an IP ID where the IPID will increase only by 1. Thus, showing the target port is closed. Explaining further: Step 1.

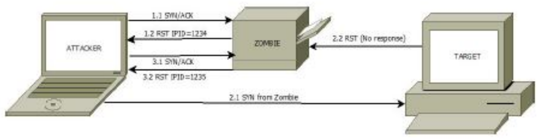


Fig. 2. Idle Stealth Port Scan for closed Port

To record IP ID of the Zombie, probe the Zombie machine

To obtain the IP ID of the zombie, the attacker is sending an SYN/ACK packet to Zombie. As a response to attacker Zombie sends back RST with IP ID.

Step 2. An Attacker is forging an SYN packet from the zombie.

The attacker sends an SYN to the target, spoofing as it is coming from the zombie. Since the target port is closed, the target sends RST to the zombie.

Step 3. To record the IP ID of the Zombie, probe the Zombie machine again.

Attacker probes the zombie again, by sending SYN/ACK, the zombie responds with RST and IP ID. Since step 1, the IP ID has been increased by 1, indicating the port is closed.

In the previous papers, idle stealth port scan performance and detection through snort have been done. We are building upon their previous work by performing additional scans namely FIN, XMAS and NULL.

Using our methodology, we will be able to find all sort of stealth port scans and we are enabling snort to send us alerts whenever an affected packet has been identified.

III. Proposed Solution

Steps involved in the process of the completion of the project:

1. Install Snort and Nmap in Kali Linux. Nmap is an open source network scanner used for port scanning and OS detection [12].

2. Develop snort modules according to the proposed methodology. This involves writing detection plugins in C and adding those files to snort source code.

3. Perform various types of port scans.

4. Write rules in snort (Snort uses a simple rule-based language to specify signatures).

For each of the scans, the proposed methodology is implemented to detect the attack. We are adding snort rules to generate alerts, whenever an attack has been performed.

IV. Methodology

As per the definitions of the FIN, XMAS and NULL scans, we should check for the flags which are being used for those attacks. While writing snort rules, our first step is to check whether the packet is TCP.

FIN scan: Once snort confirms the packet is TCP, it should check for the flags that have been set. Since snort does stateful filtering it will show alerts for any new incoming connection with only FIN flag set. Snort rule for FIN port scanning is:

alert tcp $EXTERNAL_NET any <> $HOME_NET any (msg:"FIN SCAN DETECTED"; flags:F; sid:1000001)

XMAS scan: Here, we should check whether FIN, PSH, and URG flags set. Snort rule is as follows:

alert tcp $EXTERNAL_NET any <> $HOME_NET any (msg:"XMAS SCAN DETECTED"; flags:FPU; sid:1000002)

NULL scan: As the name of the scan suggests, we are looking for any external connection with no flags set. Snort rule for NULL port scanning is: alert tcp $EXTERNAL_NET any <> $HOME_NET any (msg:"NULL SCAN DETECTED"; flags:0 ;sid:1000003)

IDLE scan: For Idle port scanning, we are considering three machines. In real time we have considered a printer as a zombie machine. It should be idle (hence the scan name), as extraneous traffic will bump up its IP ID sequence, confusing the scan logic. The lower the latency between the attacker and the zombie, and between the zombie and the target, the faster the scan will proceed. Our methodology involves, two phases:

A. Packet Capturing and Pre-Processing Phase

In this phase, we should check whether the packet is TCP. In case of TCP packet, check for SYN or RST flags and IP ID, then send the packet to the next phase, else drop the packet. Following Flow chart in figure 3 represents the complete process of this phase:

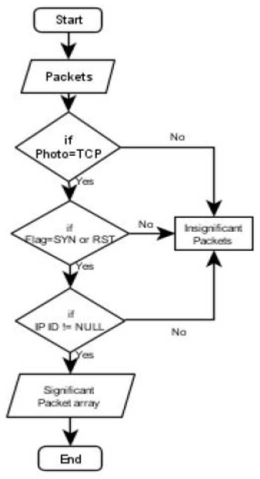


Fig. 3. Flow chart for packet capturing and preprocessing

B. Analysis and Detection Phase

In this phase, store the packet destination IP addresses of a TCP packets with SYN flag. If TCP packet has RST flag with at least one occurring SYN flag, check the destination IP address is not empty. If the destination IP address is equal to the stored destination IP address, send the packet into an idle scan attack array.

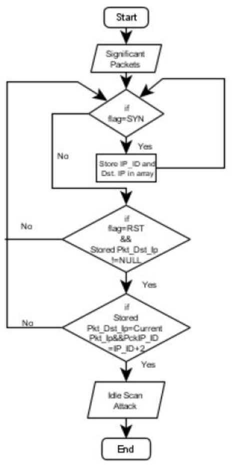

Start
Significant Packets
if flag=SYN
Yes
No
Store IP_ID and Dst. IP in array
No
if flag=RST && Stored Pkt_Dst_Ip !=NULL
Yes
No
if Stored Pkt_Dst_Ip=Current Pkt_Ip&&PckIP_ID =IP_ID+2
Yes
Idle Scan Attack
End

Fig. 4. Flow chart for analysis and detection phase

Because of these methods, using Idle Port Scan Detection Algorithm (IPSDA) we are successfully able to detect Idle port scan attacks.

Our next step is to implement these pseudo codes in the snort detection plugin. Code these using C and add these files to snort source code. Once after adding the files, edit make files.

The keyword we have used for this detection is "exist-ipid".

The rule we have created for detecting Idle port scan is as follows:

alert tcp $EXTERNAL_NET any <> $HOME_NET any (msg:"Idle scan detected"; flags:S; exist-ipid)

V. Results

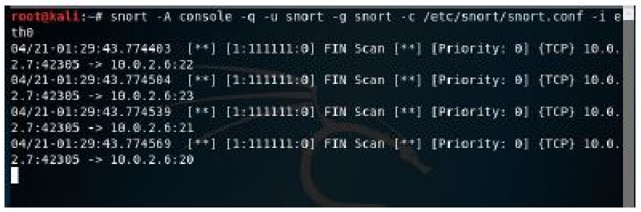
Now we have successfully finished all parts for our detection. Observe the results by performing port scan attacks.

The Command for scanning 2023 ports using FIN scan is:

nmap -sF -p20-23 Target (ip address)

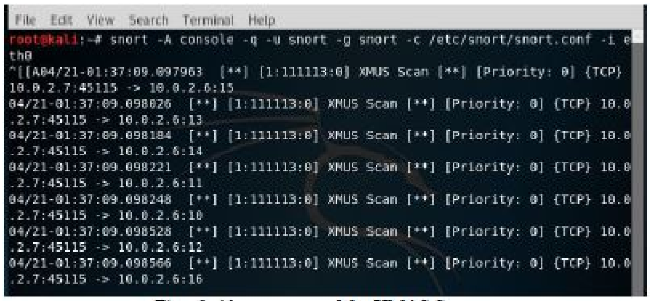
Our snort alerts are shown in figure 5 The Command for XMAS scan for ports 10-13 is:

nmap -sX -p10-13Target(ip address) and the alerts are as follows.



```
root@kali:~# snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i e
th0
04/21-01:29:43.774403  [**] [1:111111:0] FIN Scan [**] [Priority: 0] {TCP} 10.0.
2.7:42305 -> 10.0.2.6:22
04/21-01:29:43.774504  [**] [1:111111:0] FIN Scan [**] [Priority: 0] {TCP} 10.0.
2.7:42305 -> 10.0.2.6:23
04/21-01:29:43.774539  [**] [1:111111:0] FIN Scan [**] [Priority: 0] {TCP} 10.0.
2.7:42305 -> 10.0.2.6:21
04/21-01:29:43.774569  [**] [1:111111:0] FIN Scan [**] [Priority: 0] {TCP} 10.0.
2.7:42305 -> 10.0.2.6:20
```

Fig. 5. Alerts generated for FIN Scan

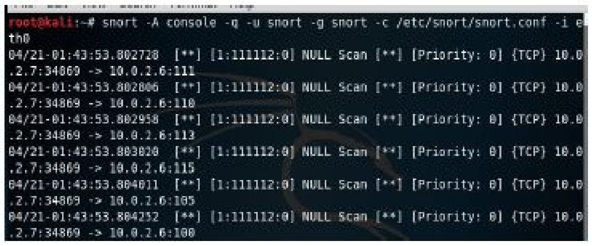


```
root@kali:~# snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i e
th0
^[[A04/21-01:37:09.097963  [**] [1:111113:0] XMUS Scan [**] [Priority: 0] {TCP}
10.0.2.7:45115 -> 10.0.2.6:15
04/21-01:37:09.098026  [**] [1:111113:0] XMUS Scan [**] [Priority: 0] {TCP} 10.0
.2.7:45115 -> 10.0.2.6:13
04/21-01:37:09.098184  [**] [1:111113:0] XMUS Scan [**] [Priority: 0] {TCP} 10.0
.2.7:45115 -> 10.0.2.6:14
04/21-01:37:09.098221  [**] [1:111113:0] XMUS Scan [**] [Priority: 0] {TCP} 10.0
.2.7:45115 -> 10.0.2.6:11
04/21-01:37:09.098248  [**] [1:111113:0] XMUS Scan [**] [Priority: 0] {TCP} 10.0
.2.7:45115 -> 10.0.2.6:10
04/21-01:37:09.098528  [**] [1:111113:0] XMUS Scan [**] [Priority: 0] {TCP} 10.0
.2.7:45115 -> 10.0.2.6:12
04/21-01:37:09.098566  [**] [1:111113:0] XMUS Scan [**] [Priority: 0] {TCP} 10.0
.2.7:45115 -> 10.0.2.6:16
```

Fig. 6. Alerts generated for XMAS Scan

The Command for NULL scan for ports 100-115 is:

nmap -sN -p100-115 Target(ip address) and alerts are

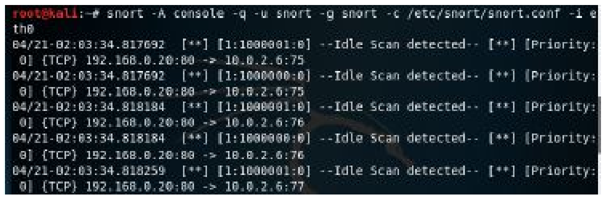


```
root@kali:~# snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i e
th0
04/21-01:43:53.802728  [**] [1:111112:0] NULL Scan [**] [Priority: 0] {TCP} 10.0
.2.7:34869 -> 10.0.2.6:111
04/21-01:43:53.802806  [**] [1:111112:0] NULL Scan [**] [Priority: 0] {TCP} 10.0
.2.7:34869 -> 10.0.2.6:110
04/21-01:43:53.802958  [**] [1:111112:0] NULL Scan [**] [Priority: 0] {TCP} 10.0
.2.7:34869 -> 10.0.2.6:113
04/21-01:43:53.803020  [**] [1:111112:0] NULL Scan [**] [Priority: 0] {TCP} 10.0
.2.7:34869 -> 10.0.2.6:115
04/21-01:43:53.804011  [**] [1:111112:0] NULL Scan [**] [Priority: 0] {TCP} 10.0
.2.7:34869 -> 10.0.2.6:105
04/21-01:43:53.804252  [**] [1:111112:0] NULL Scan [**] [Priority: 0] {TCP} 10.0
.2.7:34869 -> 10.0.2.6:100
```

Fig. 7. Alerts generated for NULL Scan

Idle scan using command: nmap -sI -Pn Zombie Target

The -Pn option prevents Nmap from sending an initial ping packet to the TARGET machine. Which would have disclosed the ATTACKER'S true address.



```
root@kali:~# snort -A console -q -u snort -g snort -c /etc/snort/snort.conf -i e
th0
04/21-02:03:34.817692  [**] [1:1000001:0] --Idle Scan detected-- [**] [Priority:
 0] {TCP} 192.168.0.20:80 -> 10.0.2.6:75
04/21-02:03:34.817692  [**] [1:1000000:0] --Idle Scan detected-- [**] [Priority:
 0] {TCP} 192.168.0.20:80 -> 10.0.2.6:75
04/21-02:03:34.818184  [**] [1:1000001:0] --Idle Scan detected-- [**] [Priority:
 0] {TCP} 192.168.0.20:80 -> 10.0.2.6:76
04/21-02:03:34.818184  [**] [1:1000000:0] --Idle Scan detected-- [**] [Priority:
 0] {TCP} 192.168.0.20:80 -> 10.0.2.6:76
04/21-02:03:34.818259  [**] [1:1000001:0] --Idle Scan detected-- [**] [Priority:
 0] {TCP} 192.168.0.20:80 -> 10.0.2.6:77
```

Fig. 8. Alerts generated for IDLE Scan

VI. Conclusion

By using Snort as an Intrusion detection system and, by designing detection plug-in for Idle port scan attack, we have seen the full ability of the snort to detect stealth port scan attacks in real time. Thus, our technique is useful in detection as well as the generation of the alerts.

In the proposed technique we have used the IP ID number to detect Idle port scan attack which is a part of IPv4 header of TCP/IP. In future, we can detect idle port scan attack in IPv6 header which does not have an IP ID field.

References

[1] R. U. Rafeeq, Intrusion detection systems with Snort: advanced IDS techniques using Snort, Apache, MySQL, PHP, and ACID. Prentice Hall Professional, 2003.

[2] R.Christopher, "Port Scanning Techniques and the Defense Against Them," SANS Institute InfoSec Reading Room October 5, 2001.

[3] R. R. Singh and D. S. Tomar "Network forensics: detection and analysis of stealth port scanning attack" International Journal of Computer Networks and Communications Security Vol. 3, N0o 2, pp. 33–42, February 2015.

[4] C. B. Lee, C. Roedel and E. Silenok, "Detection and Characterization of Port Scan Attacks,"University of California, Department of Computer Science and Engineering(2003).

[5] P. Deshpande, A. Aggarwal, S. Sharma, P. S. Kumar, A. Abraham, "Distributed port-scan attack in cloud environment", *Proc. 5th Int. Conf. Comput. Aspects Soc. Netw. (CASoN)*, pp. 27-31, 2013.

[6] C. Kao et al, *A predictive zero-day network defense using long-term port-scan recording*, IEEE Conference on Communications and Network Security (CNS), 2015.

[7] R. Gaddam, M. Nandhini, "An analysis of various snort based techniques to detect and prevent intrusions in networks proposal with code refactoring snort tool in Kali Linux environment", *2017 International Conference on Inventive Communication and Computational Technologies (ICICCT)*, pp. 10-15, 2017.

[8] A. R. Roozbahani and L. Rikhtechi, "Creating a collaborative architecture in snorts to high speed networks", *The 2nd International Conference on Computer and Automation Engineering (ICCAE)*, 2010.

[9] X. Hou, Z. Jiang, and X.Tian, "The detection and prevention for ARP Spoofing based on Snort", *International Conference on Computer Application and System Modelling* , Oct.2010.

[10] S. Patel and A. Sonker, "Internet Protocol Identification Number Based Idle Stealth Port Scan Detection Using Snort", *IEEE International Conference*, Oct.2016.

[11] B. Claypool "Stealth port scanning methods" Global Information Assurance Certification Paper – 2002.

[12] Z. Durumeric, E. Wustrow and J.A. Halderman, "Nmap network scanning: the official nmap project guide to network discovery and security scanning" Insecure, 2009.

APPENDIX

A. Advisory comments

Based on advisor's comments on our mid-term report, we made the following corrections:

1. Used LateX format for our paper.

2. Provided overview of project in Introduction section
3. In the last paragraph of the introduction we gave an overview of the organization of the rest of the paper.
4. Corrected our grammatical errors.
5. Corrected definition of IPID.
6. Referenced all figures in the text.
7. Added a related works section.
8. We gave a summary of flags are available on a TCP packet and what they are used for.
9. Methodology section is rearranged and presented in a meaningful way.
10. Moved future work to conclusion section.
11. Added a results section.