ISSN:0975-9646
Jhilam Biswas / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 14 (4) , 2023, 58-61

Study on Snort Intrusion Prevention System for Security on IP Networks and Cloud Environment

Jhilam Biswas[#1]
Department of Telecommunications Engineering
University of Maryland, College Park, MD 20742, United States
1Jhilam.biswas12@gmail.com

Abstract— Network security is one of the top concerns for any network, either a physical wired network or a virtual hybrid cloud networking system. Networks are prone to security breach from attacks hence data security is of utmost importance to any organization. In order to counter such attacks, there are a few tools available such as Firewalls, IDS and IPS. Firewalls block traffic based on access-lists defined in them. Intrusion Detection System (IDS) matches the pattern of attacks defined in them and triggers the signatures. Intrusion Prevention System (IPS) is an advanced technology that goes a step further - it performs all the functions of the IDS and additionally takes necessary actions to prevent the attack from affecting the network. Snort-IPS is an open source security tool, used widely to prevent vulnerabilities from affecting a network. This paper is a reference about Snort workflow, rule-set and how Snort detects intrusions in an IP network. The paper also talks about how Snort is deployed in a cloud environment and the appropriate countermeasures it takes to prevent future intrusions in a network.

***Keywords*— Network security, IDS, IPS, Snort, firewalls, Cloud security**

I. INTRODUCTION

Many organizations like finance companies, data service providers, retailers, health industries, research labs, universities and government organizations need a robust security tool to protect their sensitive data. A network is prone to innumerable malware attacks. Hackers, these days, use sophisticated tools, technologies, skills and abilities to intrude into a system and steal confidential information. The stolen data has the propensity to be used maliciously causing serious harm to business. Therefore, it is not only important to secure the network from threats but also prevent the threats from affecting the network. For this purpose, IPS fulfills the exact requirement. It is a new technology in the domain of network security and has gained tremendous popularity in the past decade.

Snort is a free, lightweight Network IDS (NIDS) and Network IPS (NIPS) written in C language. It was developed by Martin Roesch in 1998 and [1] since then has become a popular NIDS-NIPS used in the network security domain. The latest version of Snort is 3.0. Snort is a rule driven tool that efficiently analyzes data traffic in real time and generates alerts. It is a cross platform tool, hence can be installed on almost all computer architectures and operating systems. Essentially, Snort analyzes data traffic flow and matches the packets with a set of a predefined rule set to identify malicious or suspicious data packets.

IP Networks are most commonly prone to network probe attacks. In a network probe attack, an attacker uses a scan program like nmap, satan or mscan to collect the network's vulnerable information like IP addresses, Operating System data and hostname [1]. Snort is a very effective tool to prevent such network probe attacks.

Virtual cloud networks are used massively in many domains these days. Just like in IP networks, cloud networks are also prone to malware attacks. A robust security system can be built in a cloud environment setup by deploying a network-based Snort IPS within the cloud architecture. By utilizing the power of Snort, the cloud-networking environment can be dynamically reconfigured based on the detected attacks in real time. [2]

The paper is organized as follows: Section IIA gives an analysis of Snort rule structure, Section IIB explains the different components of Snort and Section IIC talks about where to place a Snort IPS in a network. Section III explains how Snort is used to protect an IP network from network probe attacks. Section IV explains the integration of Snort in cloud architecture to provide cloud security. Finally, the paper talks about some of the advantages and disadvantages of Snort IPS followed by the conclusion.

II. SUMMARY ON SNORT

A. Snort IDS Rule Structure

As mentioned before, Snort is a set of pre-defined rules. Every Snort rule definition consists of two logical parts: Rule header and Rule Options.

Figure 1. Snort IDS [1]

Figure 1 shows the basic structure of a Snort IDS rule. An example of a Snort header is as follows: 'alert tcp $EXTERNAL_NET any -> $HOME_NET any' [1]. The rule header portion of a Snort rule defines the criteria for matching between a rule and an incoming data packet. The header consists of the following fields:

1. The first parameter defines the type of action to be taken, for example log, alert, pass (drop), when there is a match in attack detection.
2. The next parameter is the type of the traffic the rule is looking for (for example: TCP, UDP, ICMP)
3. Source IP of the malware packets stored in the variable **$EXTERNAL_NET**
4. The destination IP address where the traffic is headed to, is stored in the variable $**HOME_NET**
5. any- This is the source or destination ports of the suspicious Traffic. It can be a single port, like for example, port 80 or arange of ports. 'any' after **$EXTERNAL_NET** is the source port number and

'any' after **$HOME_NET** is the destination port number.

6. -> Symbolically defines the direction of traffic. In the example shown above, the rule is looking for a match in the direction of traffic is from **$EXTERNAL_NET** to **$HOME_NET**

The option portion follows the header and holds a string pattern in ASCII, Hex or combination of these formats [3]. This part of the rule falls under parenthesis, and each parameter is composed of three sections- keyword, a colon and arguments in the format keyword: arguments [1] The keyword extracts information from the arguments. A semicolon separates parameters of a rule definition from each other. A single rule can have multiple parameters. Figure 2 provides an example of Snort-options.

```
(msg: "SCAN SYN FIN "; flow:stateless; flags:SF, 12; reference:arachnids,198;
classtype:attempted-recon; sid:624; rev:7);
```

Figure 2. An example of Snort options

The different parameters of Snort options are discussed below:

1. Msg – It is the alert that is triggered to the admin when a rule definition is matched to a packet. For example, in this case, the alert sent to the admin is "SCAN SYN FIN" 2. Flow – This field checks the type of traffic flow. For example, it can have values like TCP connection established, not established and stateless.
3. Flags – This field checks for various TCP flags set in the TCP header like SYN, PSH, RST, ACK, and FIN. For example, in Figure 2 shown above, the rule is looking for data packets that has both SYN and FIN packets set.
4. Reference and Class type – This field is used to point to a database to find more information about the attack. All Snort rules are classified into various categories to help the admin distinguish what type of attacks have been attempted on the network.

The next section discusses the different components of Snort.

B. Components of Snort

The flow work of Snort is composed of six parts: catching data packages, analyzing code of data, preprocessing the package, parsing the rule, detecting the engine and logging [8]. To execute this flow, Snort consists of several components that work in a particular order to detect attacks and trigger an alert from the detection system. The major components are Packet Decoder, Preprocessors, Detection Engine, Logging and Alerting System and Output modules [1]. These components are architecturally organized as shown in Figure 3.

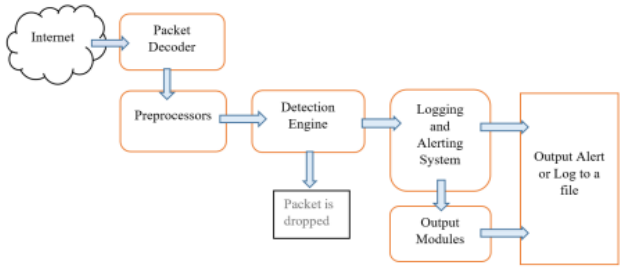

Figure 3. Organization of Snort components [4]

Data packets from the internet first enter the Packet Decoder, which fetches packets from different network interfaces and sends it to the preprocessor. Preprocessors modify the data packet in a way that is suitable for the detection engine to monitor the data and then send it to the detection engine. Detection engine is the main component of Snort where all the rule-sets exist. The rules are used into internal data structures or chains where they are compared against all incoming packets [5]. If a packet matches any rule, suitable action is taken; if not, the packet is dropped. The appropriate actions defined in the rule-set may result in logging or generating alerts. Depending on the analysis of the detection engine of a packet, this packet maybe used to log the activity or trigger an alert by the Logging and Alerting System. All logs are by default stored in the vary/log/snort directory [4]. Snort output modules perform various actions like logging to a database such as MySQL, sending SNMP traps, modifying router and firewall configurations, etc. as a prevention step to prohibit a similar attack from affecting the network at a later point in time.

C. Placement of Snort IPS within a Network

Placing a Snort IDS outside the network edge device like firewall, facing the internet, may result in many false alerts. Many of these alerts might not be a threat to the network since they are already scanned by the firewall and stopped from invading the network based on the access-lists defined in firewalls. The aim of Snort IPS is to prevent the network from legitimate attacks, which can possibly harm the network. Therefore, in most cases, a Snort IPS is installed on a device, which is in-line with the other devices on the network [6]. The device on which Snort is installed does not need an IP address, as the device just needs to sniff the network and mirror all the traffic passing through the network. In this kind of a setup, the Snort IPS is well protected from potential hackers outside of the network. Hackers cannot have easy access to the logs generated by the Snort IPS in this kind of a setup.

Speed of client honeypots can be expressed by the number of sites that can be connected and inspected in a given time period. It has significance in describing the ability of the client honeypot to identify malicious servers quickly and to safeguard client users against them. The speed of client honeypots depends on various factors such as hardware, network connection, etc. It also depends on the client honeypot implementation which means more complex the implementation, slower the honeypots are. Detection algorithms play an important role in speeding up the detection process.

III. SNORT IN IP NETWORKS

A network probe attack, as briefly described in the introduction, is essentially a mechanism to find open ports in a network, thereby making those open ports as the gateways to intrude into a network. This method is commonly called 'Port Scan' and is one of the easiest ways to invade a network. Probe attacks form the basis for performing other crucial attacks like R2L (Remote to Local), U2R (User to Root) and Denial of Service (Do's)

which can severely harm the network [1]. Snort can be a very useful tool to detect such network probes that can harm the network.

Table I lists down some of the effective Snort IDS rules to detect probe attacks.

Example Number	Snort Rule Definition
1- For HTTP	alert tcp $HOME_NET any -> $HOME_NET 80 (msg:"HTTP Scan attempted"; flow:to_server; fragoffset:0; fragbits:!D;ack:0; flags:S; window:2048; classtype: network-scan; priority:3; sid:1000003;)
2- For ICMP	alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg: "ICMP Scan Echo"; itype:8; icode:0; fragbits:!D; icmp_id:0; icmp_seq:0; classtype:attempted-recon; priority:3; sid:1000026)
3- For Domain name Service (DNS)	alert tcp $EXTERNAL_NET any -> $HOME_NET 53 (msg: "DNS request name"; flow:to_server; classtype: attempted-recon; priority:3; sid:1000034)
4- For File Transfer Protocol (FTP)	alert tcp $EXTERNAL_NET any -> $HOME_NET 21 (msg: "FTP Request connection"; flow:to_server,established; flags:AP; fragbits:D;classtype:tcp-connection; priority:4; sid:1000039)

Table I. Snort rules to detect network probe attacks [1]

In example 1, the Snort rule is written, such that it will trigger an alert when a hacker tries to scan an open HTTP service port 80. Other parameters like fragbits, flags and windows are additionally set to match packets accurately. The rule in example 2, Snort-IDS will generate an alert when an attacker sends ICMP PING packets into the network and tries to find the possible IP addresses in the network. Options like icmp_id, and icmp_seq are used in the rule for matching the packets accurately. In example 3, the Snort-IDS will generate an alert when an intruder tries to scan open DNS service port 53. TCP protocol is defined in the rule in order to increase the efficiency of detecting the packets. In example 4, the Snort-IDS will trigger an alert when the attackers try to connect to the network by sending TCP protocol packets via FTP service port number 21. The next section is dedicated to implementing Snort in cloud architecture.

IV. INTEGRATION OF SNORT IN CLOUD ARCHITECTURE TO PROVIDE SECURITY

Similar to physical IP networks, a cloud network is also exposed to malware attacks and it is important to keep cloud networks secure from such cyber-attacks. Snort in cloud is designed as a Software as a Service (SaaS) model within the cloud architecture. The Snort flow model in cloud is similar to the one discussed in Section IA, with slight variations. There are five major components to implement Snort IPS flow-model in cloud- Intrusion Detection Service Agent, Snort Daemon, Alert Interpreter, Rules Generator and Snort Database Server [2]. The service agent receives suspicious traffic alerts triggered by all virtual machines on a cloud server. Snort Daemon accumulates this data from the agent and passes it on to the alert interpreter. The alert interpreter takes care of parsing the alert. The parsed and filtered information is passed to a rules generator that generates the rules to be injected to the cloud controller to reconfigure the network and prevent future similar attacks. The database is used to store the generated rules and compare incoming packets against each of these rules.

A. Network Reconfiguration Based on Generated Alerts

Based on the different alerts generated, appropriate countermeasures are defined to mitigate the attack. Examples of these countermeasures are traffic redirection, traffic isolation, deep packet inspection, MAC address change, block port and quarantine. However, these network reconfiguration techniques may affect the network in certain ways. Intrusiveness and cost are two parameters that are used to gauge the effect of each of these countermeasures on a cloud network. Intrusiveness is defined as the negative effect that a reconfiguration has on Service Level Agreements (SLAs) [2]. Cost is used to measure the resources used and complexity needed to deploy those countermeasures. Table II lists down all of the network reconfiguration techniques against these two parameters. The results listed in Table II are based on historical experimental data.

Countermeasure	Intrusiveness	Cost
Traffic Redirection	High	High
Traffic Isolation	High	Medium
Deep Packet Inspection	High	High
Change MAC Address	Medium	Low
Change IP Address	Medium	Low
Countermeasure	**Intrusiveness**	**Cost**
Block Port	High	Low
Quarantine	High	Medium

Table II. Network Reconfiguration Actions in Cloud [2]

V. ADVANTAGES AND DISADVANTAGES OF SNORT IPS

A. Advantages

Snort is an open-source tool, which means maintaining it does not incur any licensing costs and rules monitoring traffic flow can be edited as per the needs of the organization. Snort is readily available for download from its official website. Snort is a cross-platform tool. Hence it can be installed easily on many different operating systems like Windows, FreeBSD, Linux, MacOSX to name a few. Snort is a scalable solution - rules are easy to modify, and new rules can be added to the configuration files as and when required or new attacks are detected in the network. The configuration files are centralized and can be managed from a single server where Snort is installed [6]. Network traffic monitoring with Snort is fast and efficient since it just reads the header of the raw packet to match it to the rule definitions and not the whole packet in itself. Snort provides a detailed log for what attack triggered an alert and what remediation action was taken to overcome the attack. Snort rules can be configured to isolate the parts of the network that have been attacked by malware. This helps in keeping the rest of the network unaffected by the attack.

B. Disadvantages

Even though the benefits of Snort IPS are many, it still has some limitations. Snort rules are static and lack dynamicity – Snort can only prevent traffic packets based on the defined rules and what it is told to look for in the packets and counter accordingly. If there is a new exploit, which is trying to invade the network, Snort will not notice it if there are no rules set up for this new attack. Therefore, it is imperative to keep the Snort configuration files up-to-date and maintain it regularly, which means Snort includes an additional administrative overhead. Another disadvantage of Snort is that it can be placed only behind the firewall, inside the network [7]. Placing a Snort IDS facing the internet can result in many false positive alerts. Snort requires other tools like Apache, MySQL, and PHP to be pre-installed before installing it on any operating system [8].

CONCLUSIONS

The above paper gives a brief overview of the Snort IPS and how it can help to detect and prevent attacks real-time in IP and cloud networks. If configured appropriately, it can help secure the network very efficiently based on the defined rule-set. Although Snort is an efficient tool to detect malware invading a network and preventing them, it comes with the added challenge of writing good rules and maintaining a good rule-set. It also provides a systematic way of logging all alerts, which helps to protect the network from future attacks. Another major drawback of the Snort IPS is that it is not intuitive and lacks automation [4]. Attackers are developing newer techniques to invade the network and steal information. Therefore, it is crucial to know in advance what kind of threats the network could possibly be exposed to. Currently, there have been researches going on to make the Snort IPS intelligent and intuitive, which will provide a robust and full proof protection to the network infrastructure from intruders.

ACKNOWLEDGEMENT

Sincere thanks to the professors of the Department of Telecommunications for providing excellent facilities to carry out the research study and gather valuable knowledge on IDS and IPS systems.

REFERENCES

[1] N. Khamphakdee, N. Benjamas, and S. Saiyod, “Improving intrusion detection system based on Snort rules for network probe attack detection,” *2014 2nd Int. Conf. Inf. Commun. Technol. ICoICT 2014*, pp. 69–74, 2014.

[2] T. Xing, D. Huang, L. Xu, C. J. Chung, and P. Khatkar, “SnortFlow: A OpenFlow-based intrusion prevention system in cloud environment,” *Proc. - 2013 2nd GENI Res. Educ. Exp. Work. GREE 2013*, pp. 89–92, 2013.

[3] H. Li and D. Liu, “Research on intelligent intrusionprevention systems based on Snort,” 2010 *Int. Conf. Comput. Mechatronics, Control Electron. Eng. C. 2010*, vol. 1, pp. 251– 253, 2010.

[4] S. Khamitkar, “Network Intrusion Detection using SNORT,” *Int. J. Eng. Res. Appl.*, vol. vol.2, no. Issue 2, pp. 1288–1296, 2012.

[5] H. Alnabulsi, M. R. Islam, and Q. Mamun, “Detecting SQL injection attacks using SNORT IDS,” *Asia-Pacific World Congr. Comput. Sci. Eng. APWC CSE 2014*, 2014.

[6] A. Bair, “Snort Management System : Managing Multiple Snort Instances on Many Systems,” pp. 1–7, 2006.

[7] C. N. Modi, D. R. Patel, A. Patel, and M. Rajarajan, “Integrating Signature Apriori based Network Intrusion Detection System (NIDS) in Cloud Computing,” *Procedia Technol.*, vol. 6, pp. 905–912, 2012.

[8] Z. Zhou, “The study on network intrusion detection system of Snort,” *2010 Int. Conf. Netw. Digit. Soc.*, pp. 194–196, 2010.