LAB 4: SNORT

Snort is an open source network intrusion prevention system, capable of performing real-time traffic analysis and packet logging on IP networks. It can perform protocol analysis, search/match content, and detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, and much more.

Snort and the related documentation can be found at the following link: https://www.snort.org/

Snort can be run in three modes:

1. Sniffer mode: which simply reads the packets off of the network and displays them for you in a continuous stream on the console.
2. Packet logged mode: which logs the packets to disk.
3. Network Intrusion Detection system (NIDS): which performs detection and analysis on network traffic. This is the most complex and configurable mode.

In this lab, you will use the NIDS mode.

Please use Snort on the Int-rtr VM for this assignment.

Your configuration file is **snort.conf**, which can be found at **/etc/snort/etc**. Make sure you are using the snort.conf at this exact location, since there are files named snort.conf at very similarly-named locations. An example command line for snort used in NIDS mode is shown below:

snort -dev -A test -c <configuration file, i.e. snort.conf> -i eth0

(-dev instructs packet to display the packet data as well as headers.)

This configuration file will include the rules configured for each packet to decide if an action should be triggered based on the rule type.

The output is stored in the **alert** file (if you use -A test) which is located at **/var/log/snort** and also displayed on the screen in the following format:

[**] [116:56:1] (snort_decoder): T/TCP Detected [**]

- The first number is the **Generator ID**; this tells the user what component of Snort generated this alert. For a list of GIDs, please read etc/generators in the Snort source. In this case, we know that this event came from the “decode” (116) component of Snort.
- The second number is the **Snort ID** (sometimes referred to as Signature ID). To learn about preprocessor SIDs, please see https://www.snort.org/rule_docs. Rule-based SIDs are written directly into the rules with the sid option.

- The third number is the **revision ID**. This number is primarily used when writing signatures, as each rendition of the rule should increment this number with the rev option.

There are a number of alert modes which can be used using '–A' to append it to the command. We will be making use of the **fast, full and test**.

Please use the PCAP file for this assignment. It can be found at **/home/student/snort_src/InfectedPcaps/infected.pcap**. To read a pcap file using Snort, you can use one of the following options:
$ sudo snort -r <file>
$ sudo snort --pcap-single=<file>
Please wait for the message "Snort exiting" before reading the results.

SUBMIT FOR THE ASSIGNMENT:

Answer the following questions using the alert log file provided. Please provide screenshots wherever necessary.

1. List the alerts (from the alerts) and list the corresponding Generator ID, Snort ID and Revision ID of each alert and their significance. If an alert ID repeats multiple times only include it once. (20 points)
2. The alert file contains the output when the file was run using the "–A test" option. This will display the packet numbers of the corresponding packets that triggered alerts. Use Wireshark and locate these packets. List the source and destination IP address, source and destination port numbers and protocol used for each packet. (15 points)
3. Look up the meaning of the alerts and explain which alert you think is actually the one that identifies a likely malware instillation attempt. [15 points]
4. There were likely many likely false positive alerts that repeat many times and consumed time in your analysis. Describe your recommendation for reducing these false positives so that they would not consume analysis time in the future. Also include any potential dangers in your proposed method. [10 points]

Wireshark Exercises:

5. Use a filter and list all the DNS queries and the resolved IP addresses. Include the filter in the lab write up. (15 points)
6. There were HTTP sessions established to download 2 java applets. What were the names of the two .jar files that implemented these applets? (10 points)
7. As part of the infection, a malicious executable file was downloaded onto the client's computer. What was the file's MD5 hash? Hint: It ends on "91ed". (10 points)
8. Which browser is being used by the client? (5 points)