**LAB 3: 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, content searching/matching, and can be used to 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. You will use the **snort.conf** configuration file which is located at /etc/snort.

An example command line for snort used in NIDS mode is shown below:

*snort -dev <IP address and subnet mask, i.e. 192.168.1.0/24> -c <configuration file, i.e. snort.conf>*

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

Where snort.conf is the name of your snort configuration file. This configuration file will include the rules configured in the snort.conf file to each packet to decide if an action should be triggered based on the rule type in the file should be taken.

**The output is stored in the alert file 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). For a list of preprocessor SIDs, please see etc/gen-msg.map. 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 the ‘–A’ to append it to the command. We will be making use of the **fast, full and test**.

Reading a pcap file using Snort:

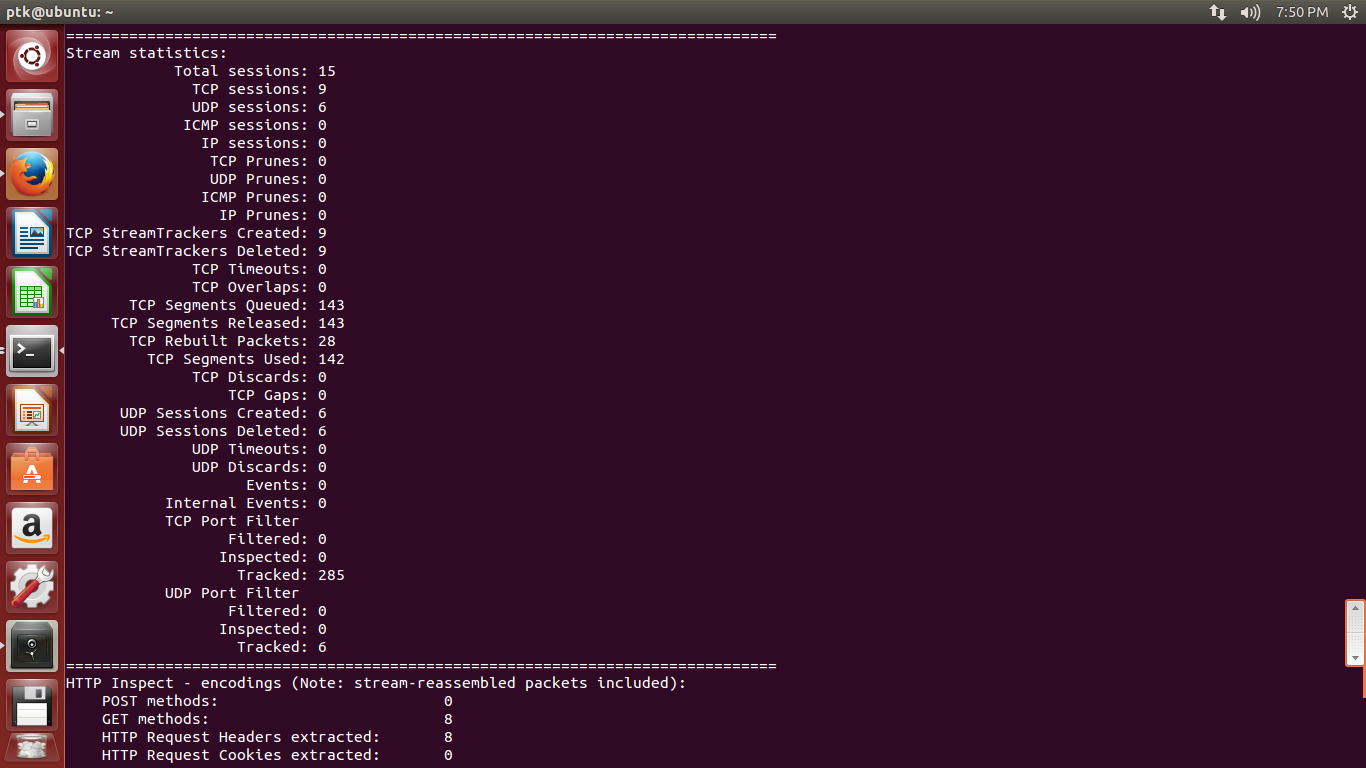
To read the pcap file you can use one of the following options:

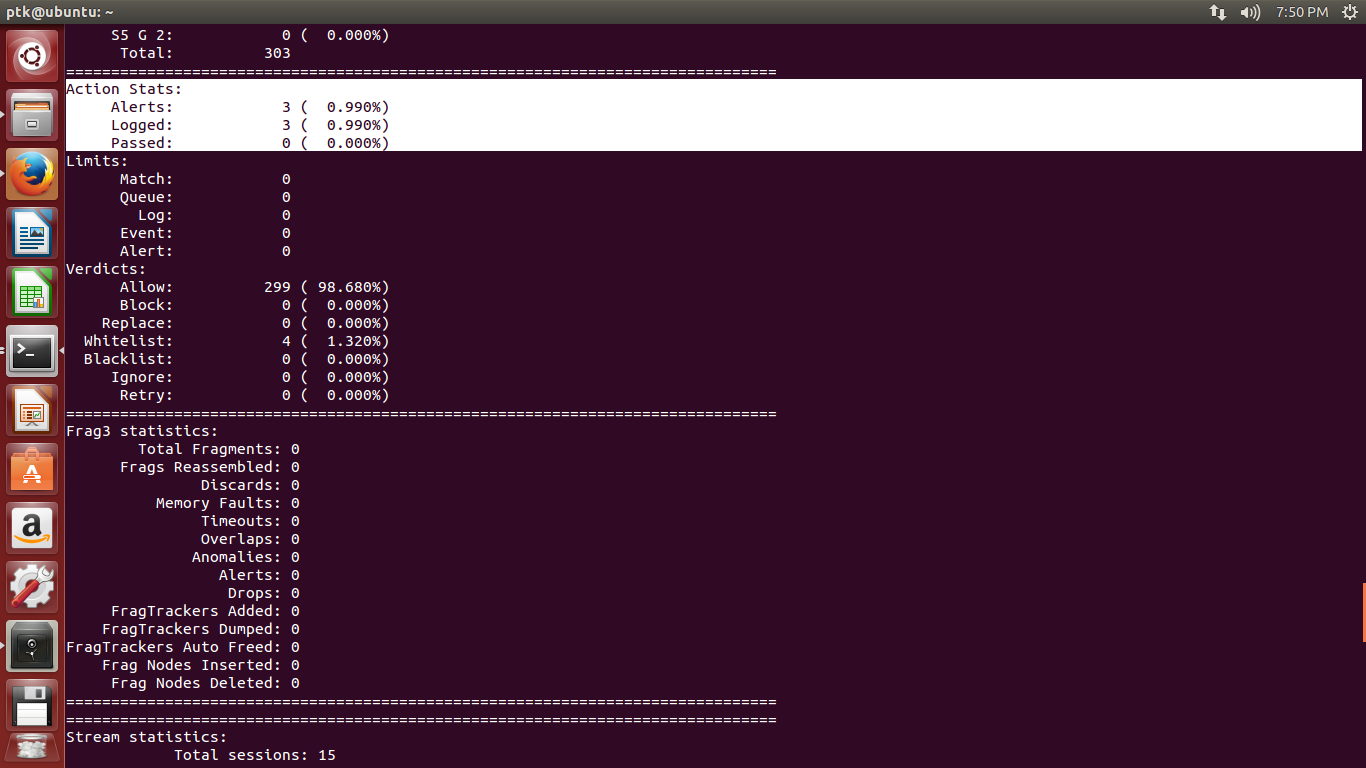
*$ snort -r <file>*

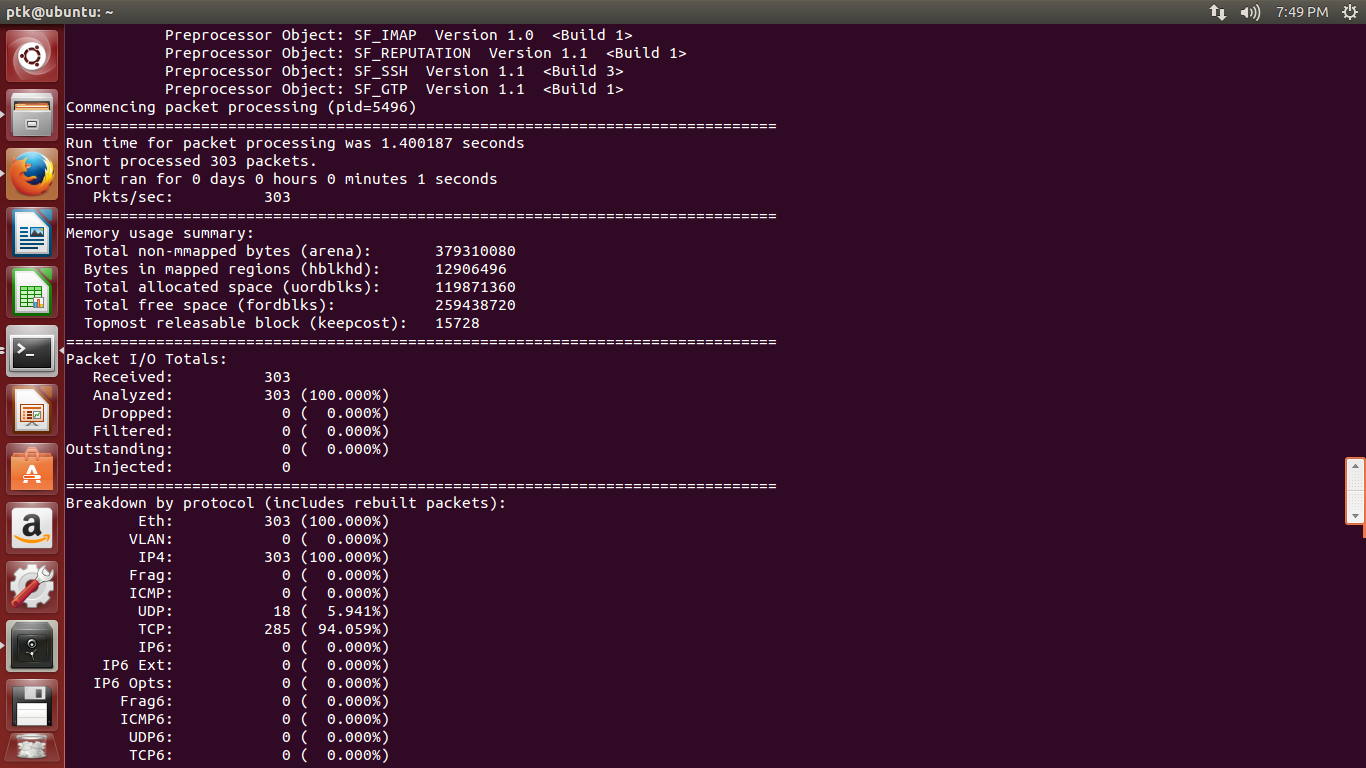
*$ snort --pcap-single=<file>*

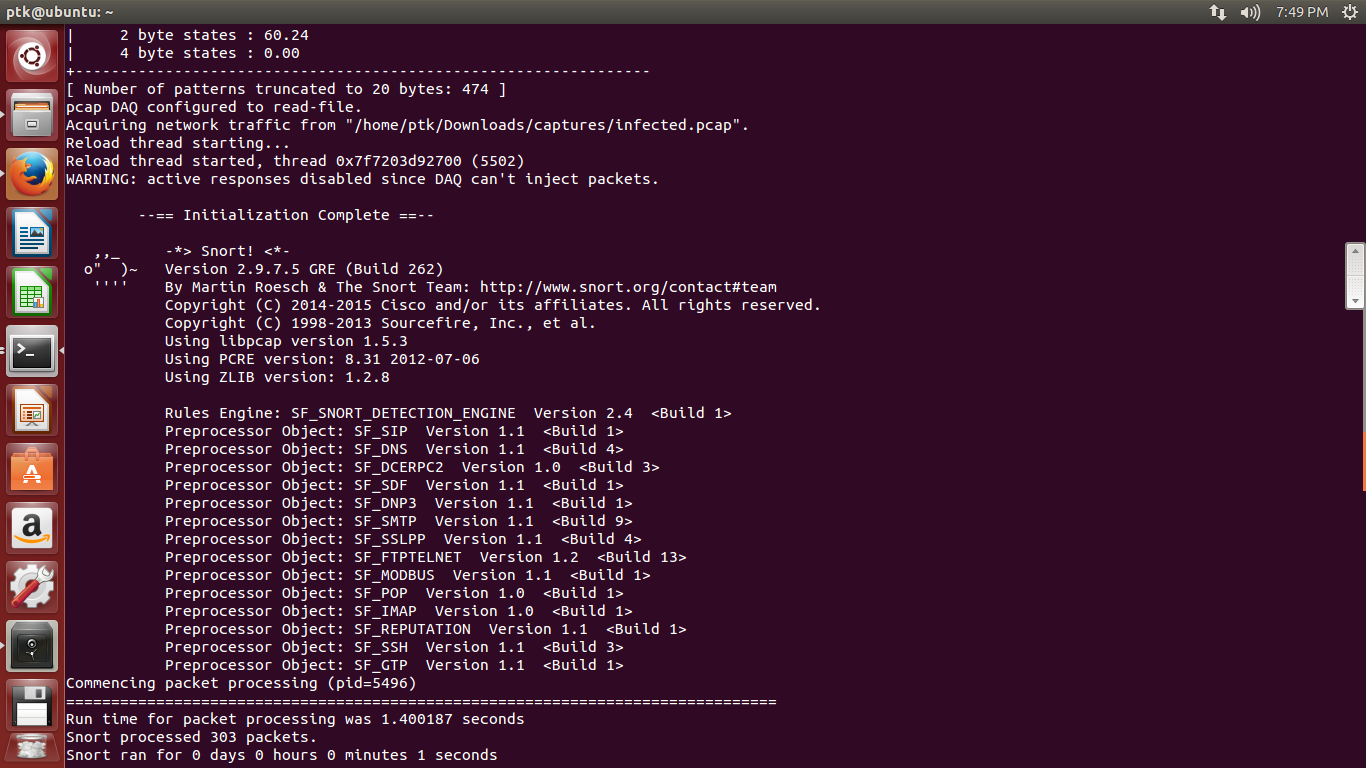
**The snort installed in the VLAB is outdated. It will not show any alerts and hence we cannot run snort on the pcap file. I have run snort on the pcap file and have found 3 alerts. I have screenshots in this document that shows the alerts.**

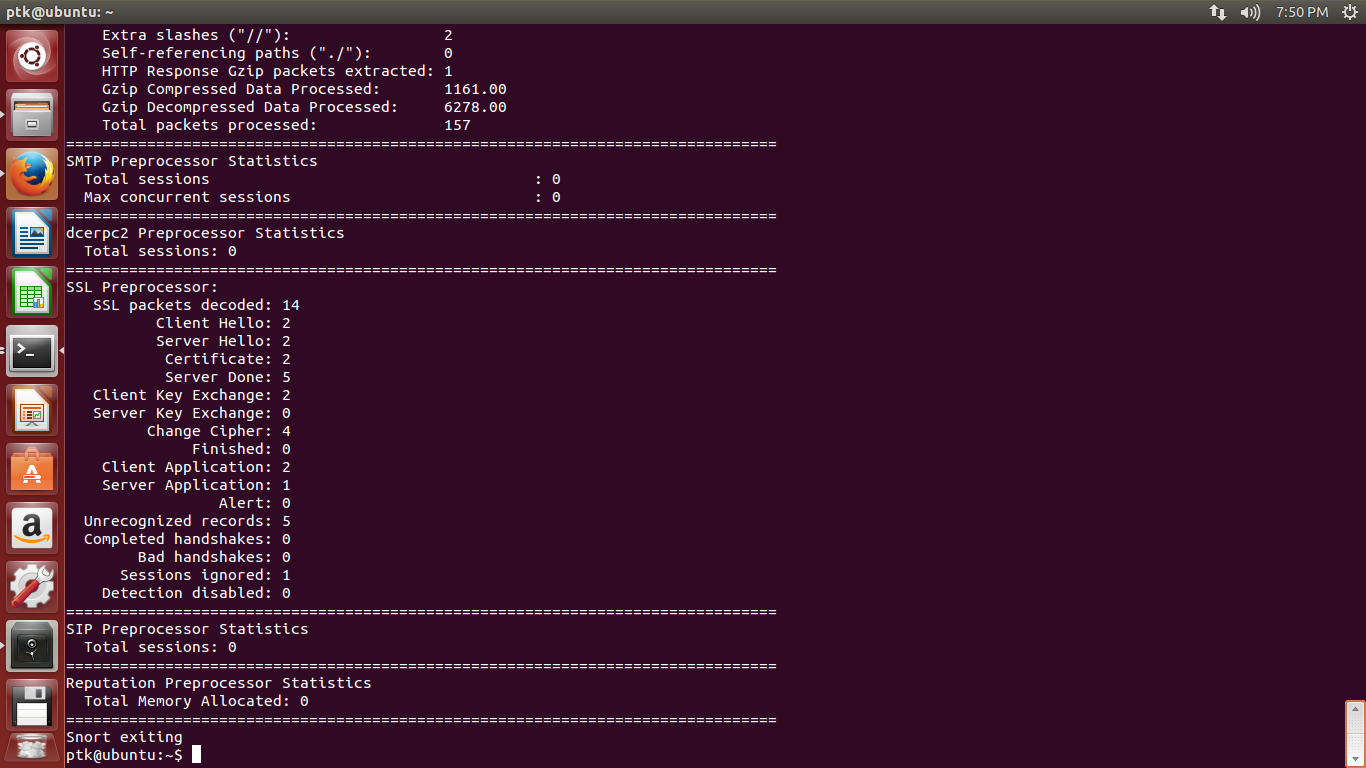
**Also, I have attached the logged alert file along with this document. Make use of that log file to answer the questions asked in the lab.**











**Answer the following Questions using the screenshots and the alert log file provided. 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. (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)

**Wireshark Exercises:**

1. Use a filter and list all the DNS queries and the resolved IP addresses. Include the filter in the lab write up. (15 points)

2. 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)

3. 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)

4. Which browser is being used by the client? (5 points)