**SNORT - PART 1**

Snort is a widely used and well-respected open-source system for detecting and preventing network intrusions. It is particularly noteworthy for its ability to actively monitor network traffic in real-time and identify potential security threats and vulnerabilities. Developed by Sourcefire and now maintained by Cisco, Snort uses a rule-based approach to create custom or pre-defined signatures that detect specific types of malicious activity. This system is highly versatile and can be deployed in various network architectures, with its protocol analysis used to decode multiple network protocols for detecting suspicious behavior. Snort's alerting system notifies administrators immediately of potential security breaches, making it an invaluable tool for intrusion detection, network monitoring, forensic analysis, and regulatory compliance adherence. With its community-driven development and regular updates, Snort remains a robust and adaptable tool in the fight against evolving cyber threats.

In this project, I will use VMware Fusion on my machine and install Snort in Ubuntu, and I will use Kali to scan the Ubuntu ports and try to connect with Ubuntu to see how Snort reacts to these requests. Also, we will review a few snort rules to see how to snort rules initiate and how this work.

Before installing Snort, we will run these commands:

- **sudo apt update**
- **sudo apt upgrade in Ubuntu**

Because sudo apt update ensures that your system's package lists are up to date, it fetches information about available packages and their versions from repositories. This step ensures you get the latest versions when installing new software or dependencies.
And sudo apt upgrade upgrades the installed packages to their latest versions. This step is essential for security reasons as it installs the latest security patches and bug fixes, reducing vulnerabilities that could affect the installation and operation of Snort.
Now, to install snort in Ubuntu, enter the command here -

- **sudo apt-get install snort -y**

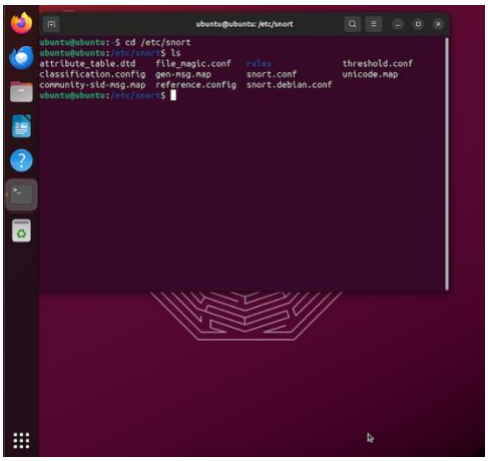
In the middle of installing the snort, it will ask you to set up the IP address of your local machine. It means Snort will look after the IP address and notify us if any rules are triggered.

In this case, I set up my IP 192.168.1.103/16.

Now, let's see what is included in the snort application. In Ubuntu, snort has been installed in the root.

You can find the folder by typing

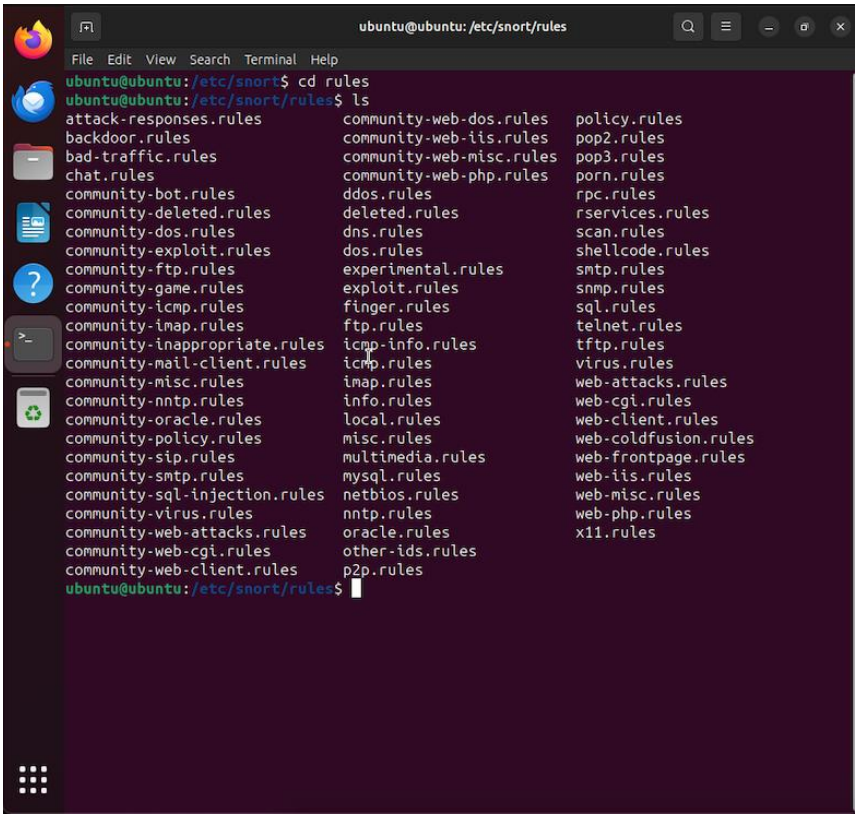
- **cd /etc/snort**




We can see files and folders in the snort folder, but the first file I want to look at is in the "rules" folder.

Go to the rules folder by typing,

- **cd rules**



By opening this, you will see a lot of files with extensions in the .rules name. These .rules files contain the detection rules for identifying and responding to specific network traffic patterns or behaviors. These rules are written in Snort's rule language and play a crucial role in Snort's intrusion detection and prevention capabilities.
Each .rules file typically focuses on a specific aspect of network traffic or a particular protocol, containing rules tailored to detect and respond to behaviors or characteristics associated with that protocol or category of network activity.

Let's overview a few rules files to explain these files clearly. First, I want to overview -

## icmp.rules

To view icmp.rules in snort, type the below command in the terminal -

- **cat /etc/snort/rules/icmp.rules**

The Snort file icmp.rules contain rules to detect and respond to ICMP network traffic. ICMP is a protocol for network-related communications, including error reporting, diagnostic messages, and network management.
These rules within icmp.rules are written in a specific syntax that Snort uses to identify and respond to particular patterns or characteristics in ICMP packets. They define criteria that, when met, trigger alerts or actions within Snort.

The rules in icmp.rules include conditions to detect:

- Distinctive patterns are linked with identified weaknesses or attacks that use ICMP packets.
- Patterns of unusual or suspicious ICMP traffic may indicate potential security threats, such as ping sweeps or ICMP-based attacks.
- Signatures may indicate potentially malicious activities such as ICMP-based DoS attacks or ICMP redirect attempts.
- ICMP packets can indicate network anomalies or misconfigurations.

## community-ftp.rules

To view community-ftp.rules in snort, type the below command in the terminal -

- **cat /etc/snort/rules/community-ftp.rules**

The file named community-ftp.rules in the Snort software usually consists of rules created by the Snort community to identify and react to FTP (File Transfer Protocol) network traffic. FTP is a widely used network protocol that enables file transfer between a client and a server on a computer network. Snort's rule language is used in community-ftp.rules to identify patterns or characteristics in FTP packets, which trigger alerts or actions when matched.
The rules within community-ftp.rules include conditions to detect various aspects related to FTP traffic, such as:

- Detection of specific FTP commands or sequences, like USER, PASS, LIST, RETR, STOR, etc., used by clients to interact with FTP servers.
- Patterns indicating unusual or suspicious behavior within FTP traffic could indicate potential security threats or attacks targeting FTP servers or clients.
- Signatures designed to identify authentication attempts, including failed login attempts, brute-force attacks, or unauthorized access to FTP servers.
- Conditions to identify specific file transfer actions, including uploads, downloads, or modifications of files via FTP.

These rules, created by the Snort community, address various aspects of FTP traffic, including vulnerabilities, attack patterns, and potentially malicious behavior.

Now, I will add a few custom rules in local.rules file to test the snort application. To edit the local.rules file type -

- **sudo nano local.rules**

ensure you are in the - **/snort/rules directory.**

**alert icmp $EXTERNAL_NET any -> $HOME_NET any (msg: “ubuntu message”; sid: 5889; rev: 1;)**

**alert tcp any any -> $HOME_NET 21 (msg: “FTP Attempted”; sid: 60001; rev: 1;)**

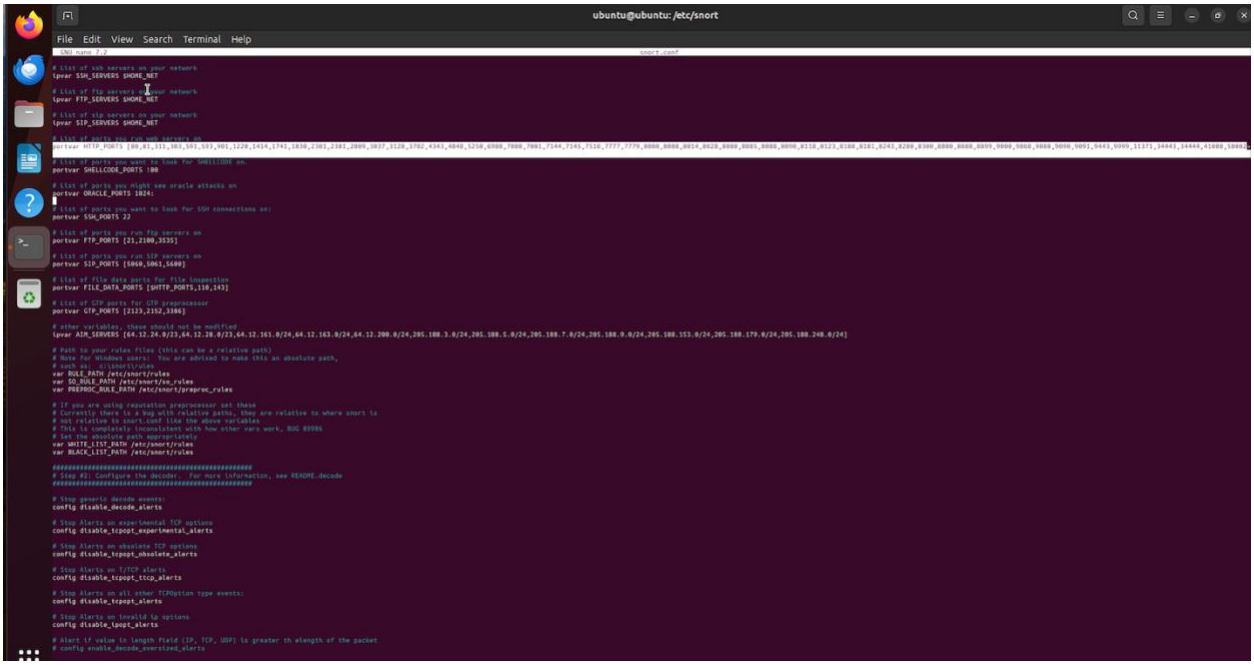
**alert tcp any any -> $HOME_NET 22 (msg: “SSH Attempted”; sid: 60001; rev: 1;)**

After saving the file, we will check the rules have been saved successfully in the configuration file. Type this command here –

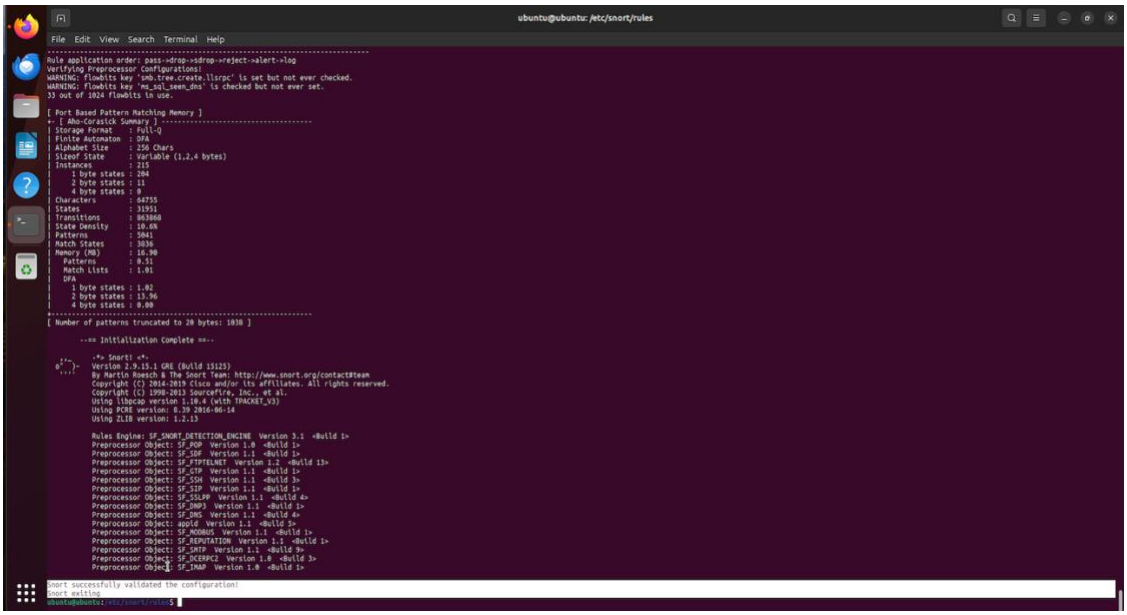
- **sudo snort -T -c /etc/snort/snort.conf**

Now you may ask what is **snort.conf**?

The snort.conf file in Snort serves as the core configuration file, governing the entire operation of the intrusion detection system. This critical file orchestrates Snort's behavior, dictating what network traffic to monitor, how to process it, and what actions to take upon detecting specific patterns or behaviors. Within this file, settings encompass a range of crucial parameters: defining the network interfaces to watch, configuring preprocessors that handle packet preprocessing tasks, specifying the detection rules and their locations, setting up logging and alerting mechanisms, configuring output plugins for handling alerts and logs, optimizing performance through memory allocation and other tuning options, and a multitude of other configurations. The snort.conf file acts as the centralized control hub, allowing administrators to fine-tune Snort's functionality to align with their environment's specific security needs, network architecture, and threat landscape. It's a pivotal resource for tailoring Snort's ability to detect and respond to potential security threats effectively.



If you successfully saved the rules in the local.rules file, and snort's configuration file successfully saved the configuration. You will see this message bellow –




After successfully saving the configuration, I will start the Snort service to start our machine firewall. To start the service, type the following command in the terminal -

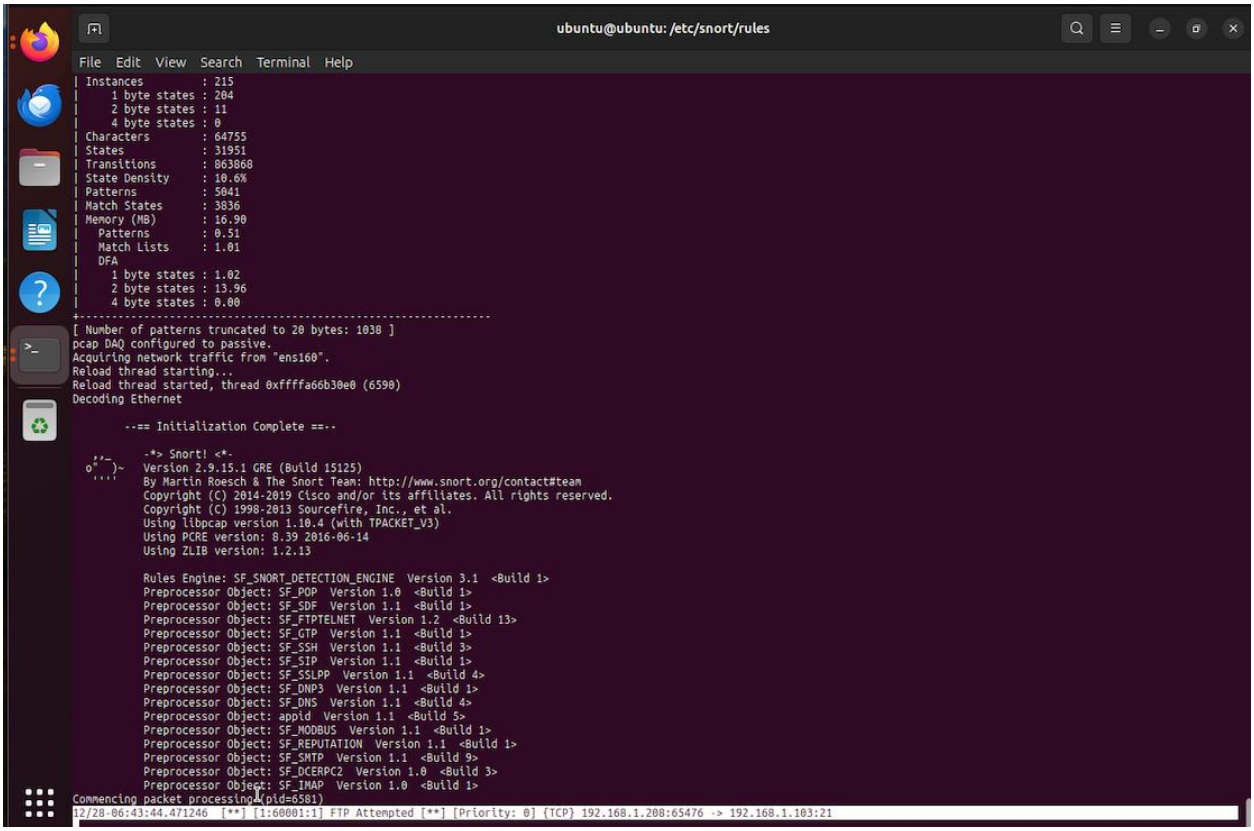
- **sudo snort -A console -c /etc/snort/snort.conf**

Now, I will start the Kali machine and will try to send FTP connection request from Kali to Ubuntu. To send an FTP request, enter the following command in the terminal -

- **ftp "IP address"**

In my case, my IP address will be **192.168.1.103**

Here, you can see my snort rules have triggered and captured the connection request.




**12/28-06:43:44.471246 [**] [1:60001:1] FTP Attempted [**] [Priority: 0] {TCP} 192.168.1.208:65476 -> 192.168.1.103:21.**

This log entry suggests that a system with the IP address 192.168.1.208 initiated an FTP-related action (as detected by the system's rules) toward another device with the IP address 192.168.1.103 on port 21, which is commonly used for FTP control connections.

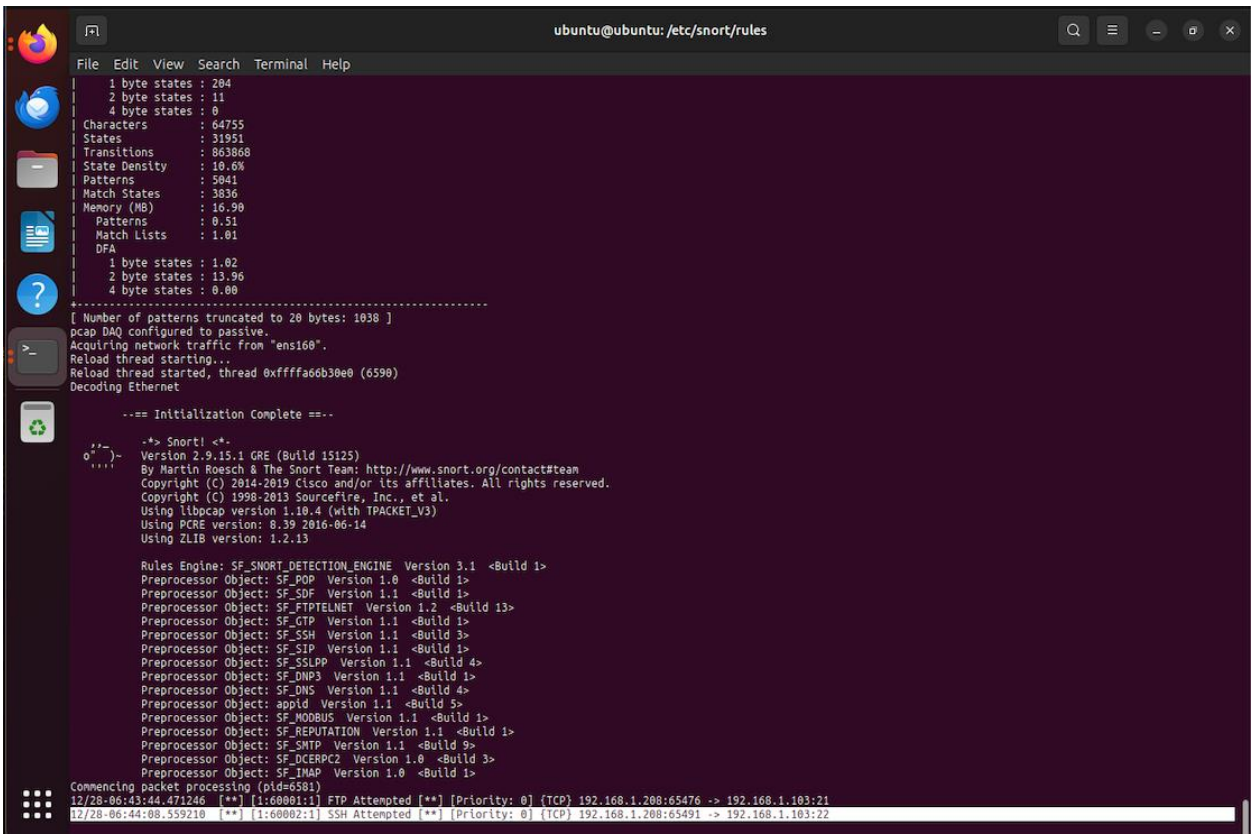
Let's break it down:

- **12/28-06:43:44.471246:** This part is called Timestamp. Indicates the date and time of the event when this entry was logged. In this case, the date is December 28th, and the time is 06:43:44.471246.
- **[1:60001:1]:** This is a rule ID from the detection system's rule set. Each rule typically has an ID associated with it, helping to identify the specific rule that triggered the alert.
- **FTP Attempted:** This is the description or message associated with the rule that fired. It signifies that the system detected an attempt related to FTP (File Transfer Protocol).
- **[Priority: 0]:** Indicates the priority level assigned to this alert. In this case, it's set to zero, which might denote a low-priority alert.
- **{TCP}:** Specifies the protocol used for this network activity, which is TCP (Transmission Control Protocol) in this instance.
- **192.168.1.208:65476:** This represents the network connection's source IP address and port number attempting the action.
- **192.168.1.103:21:** Denotes the destination IP address (likely a server or device) and its associated port number, 21, the default FTP port (control connection).

I will try to send an SSH connection from Kali to the Ubuntu machine again. To send an SSH connection request, write the command below to the Kali terminal.

**ssh 192.168.1.103**

Here, you can see my snort in Ubuntu rules have triggered again and captured the connection request.




**12/28-06:44:08.559210 [**] [1:60002:1] SSH Attempted [**] [Priority: 0] {TCP} 192.168.1.208:65491 -> 192.168.1.103:22.**

This log entry appears to be another event captured by Snort. This log entry indicates that a device with the IP address 192.168.1.208 attempted an SSH-related action (as detected by the system's rules) towards another device at IP address 192.168.1.103 on port 22, commonly used for SSH connections.

Let's break down the information provided:

- **12/28-06:44:08.559210:** Indicates the date and time of the logged event. In this case, it occurred on December 28th at 06:44:08.559210.
- **[1:60002:1]:** This is the rule ID associated with the event, serving as a unique identifier for the specific rule that triggered this alert.
- **SSH Attempted:** The message associated with the rule that was triggered indicates an attempt related to SSH (Secure Shell).
- **[Priority: 0]:** Specifies the priority level assigned to this alert. Here, it's set to zero, suggesting a low-priority alert.
- **{TCP}:** Denotes the protocol used for this network activity: TCP (Transmission Control Protocol).
- **192.168.1.208:65491:** Represents the network connection's source IP address and port number that initiated the action.
- **192.168.1.103:22:** Indicates the destination IP address (potentially an SSH server) and its associated port number, 22, the default SSH connection port.