Report : SNORT

Introduction :

SNORT is a powerful open-source intrusion detection system (IDS) and intrusion prevention system (IPS) that provides real-time network traffic analysis and data packet logging. SNORT uses a rule-based language that combines anomaly, protocol, and signature inspection methods to detect potentially malicious activity.

Using SNORT, network admins can spot denial-of-service (DoS) attacks and distributed DoS (DDoS) attacks, Common Gateway Interface (CGI) attacks, buffer overflows, and stealth port scans. SNORT creates a series of rules that define malicious network activity, identify malicious packets, and send alerts to users.

Objective :

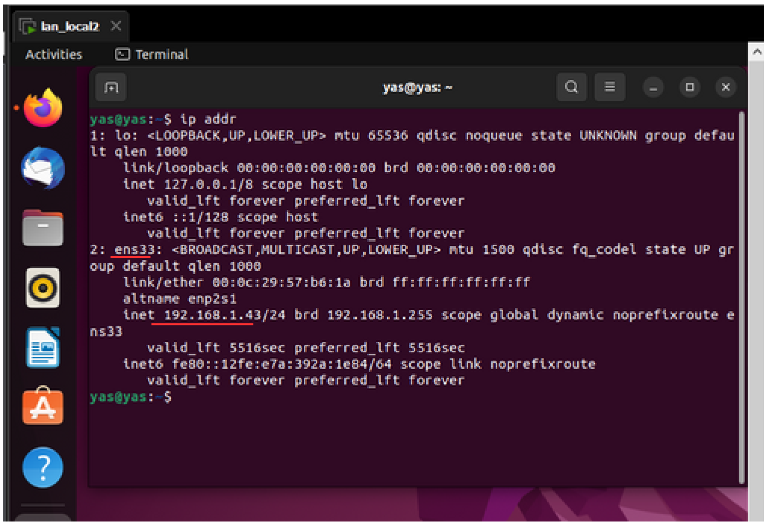
The objective of this lab is to create a hands-on experience with Snort, an intrusion detection system, using two local machines that are part of the same LAN segment specifically the "Green" network segment, and their connectivity is facilitated through IPCop, (PCop acts as a firewall and router).

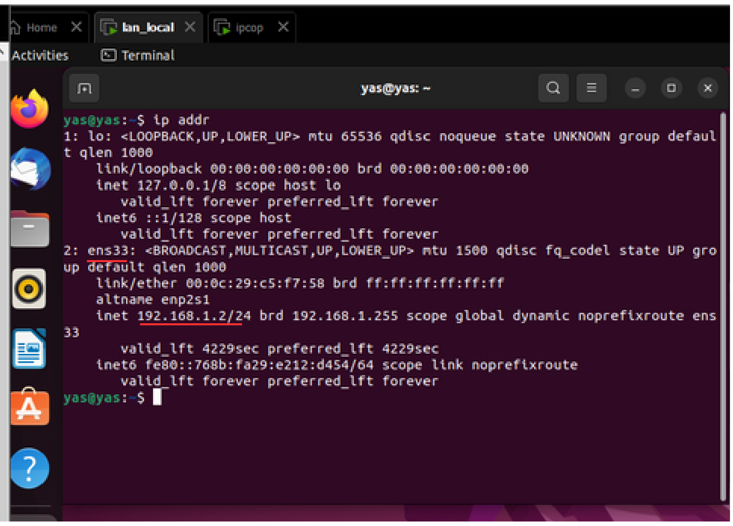
One machine will act as the attacker, and the other machine will serve as the victim. The focus of this lab is to familiarize ourselves with Snort's capabilities for detecting and analyzing network-based attacks.

Even though it is installed on a machine (victim machine) within the same network. It showcases the capability of Snort to act as an internal NIDS, providing visibility into potential security threats within the local network

Configuration :

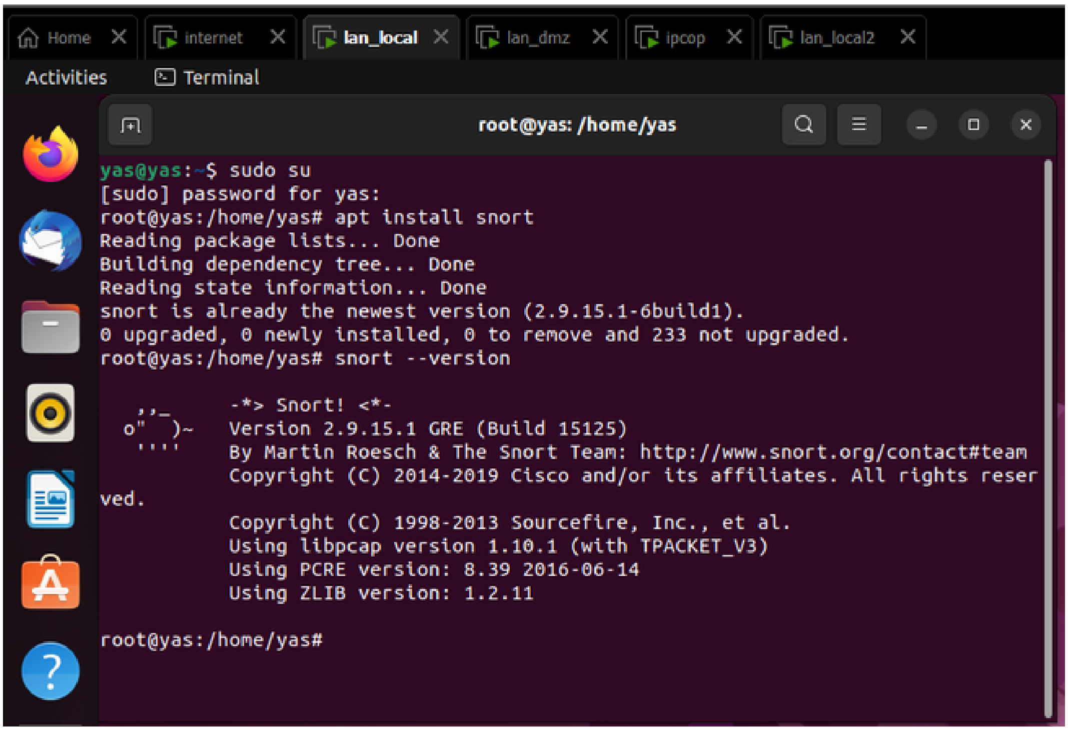
- Local machines conf :



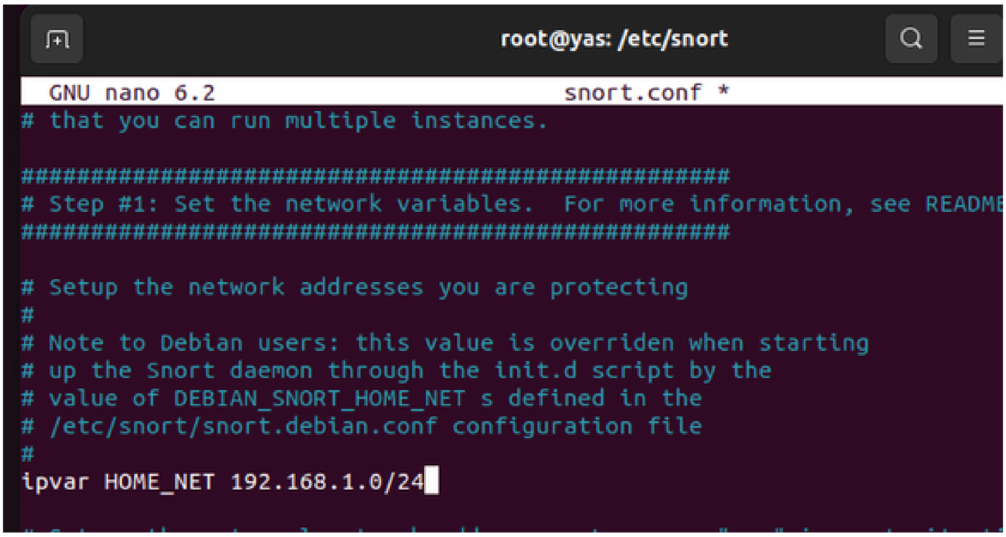


- Install Snort :

We can install Snort simply using : **sudo apt install snort**



- Configure Snort :



Snort configuration file is /etc/snort/snort.conf, this is a big configuration file, we define the specific local network that Snort will monitor, by changing the ipvar HOME_NET from "**any**" to our local network "**192.168.1.0/24**"

- Start Snort :

The command **snort -A console -q -c /etc/snort/snort.conf -i ens33** is used to start Snort with specific options and configurations.

-**A console** : Specifies the output mode as console, which means Snort alerts will be displayed in the console/terminal.
-**i ens33** : Sets the network interface to monitor for network traffic

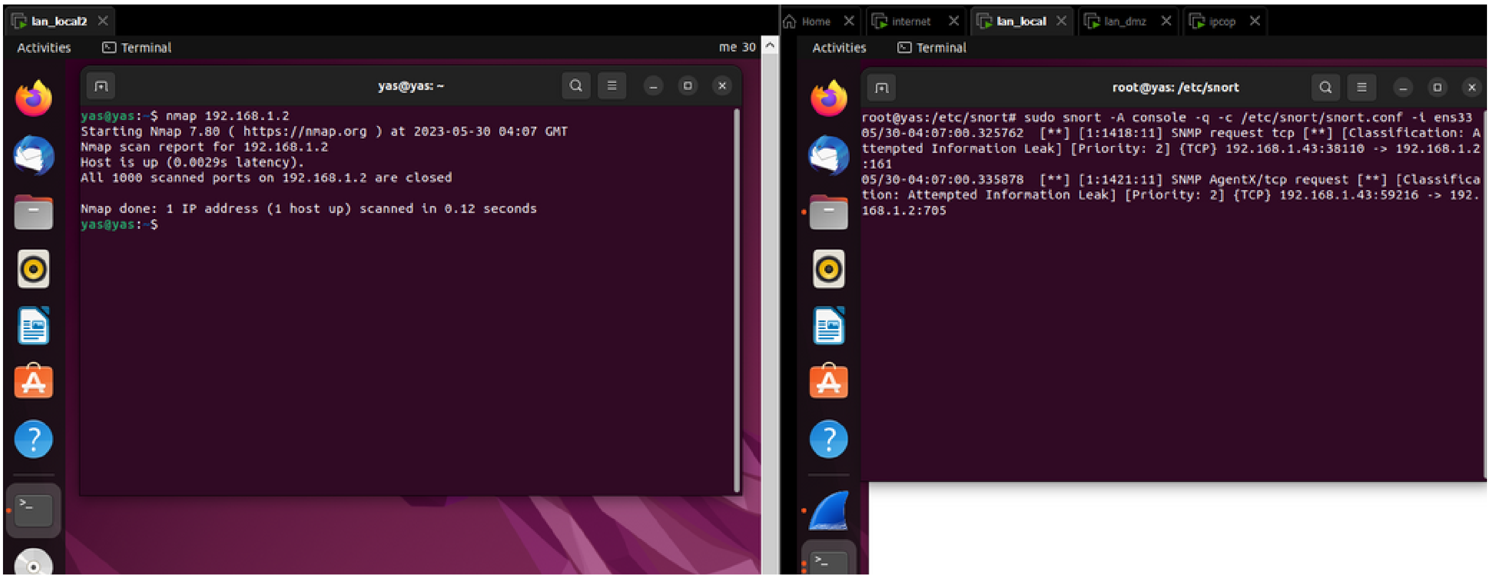
Now snort will start monitoring the network interface ens33, apply the rules and configurations specified in the snort.conf file, and display alerts in the console in a quiet mode without additional informational messages.

Default Rules :

By default, Snort comes with a set of rules that cover a wide range of network threats and attacks. These rules are designed to detect known signatures and patterns associated with various attacks and exploits.

- ## Nmap :

When it comes to basic port scanning, Snort's default rule set includes rules that can potentially detect port scanning activities



When Nmap scans are detected, Snort generates alerts indicating the presence of a potential port scanning attempt

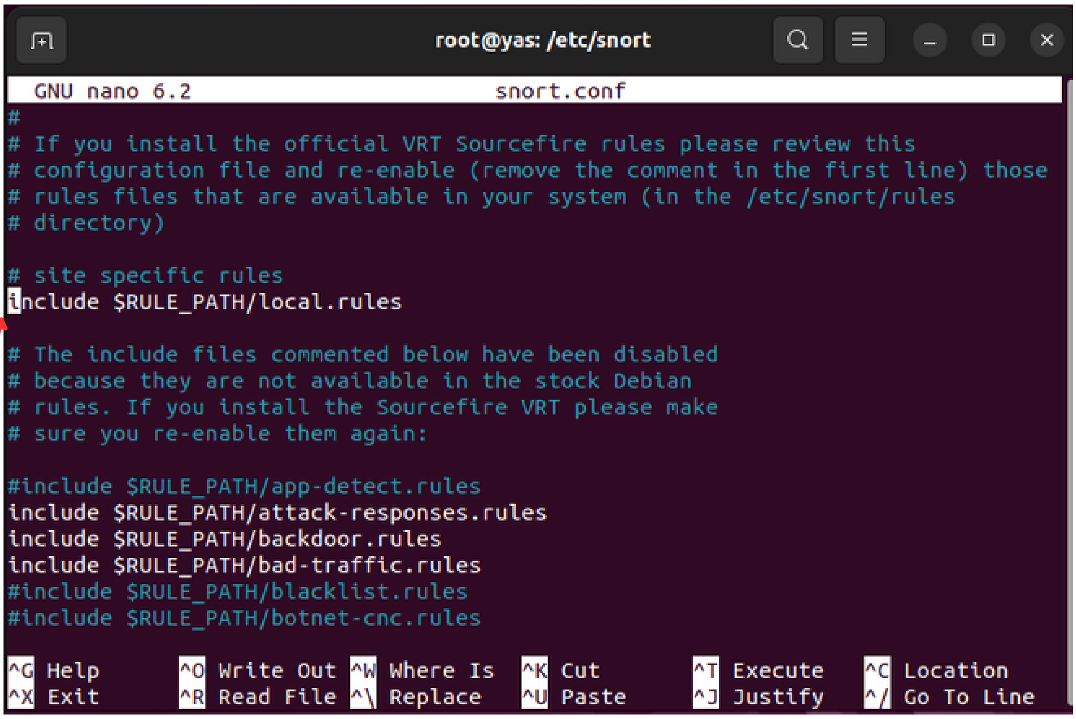
Custom Rules :

Snort offers the flexibility to create our own rules tailored to our specific network environment and security requirements. Custom rules allow us to detect and respond to unique threats, vulnerabilities, or attack patterns that may not be covered by the default rule set.

By crafting custom rules, we can define specific criteria, including specific protocols, ports, content patterns, or behaviors that we want Snort to monitor and alert on. This level of customization enables us to address the specific security concerns and characteristics of our network, providing enhanced visibility and proactive threat detection.

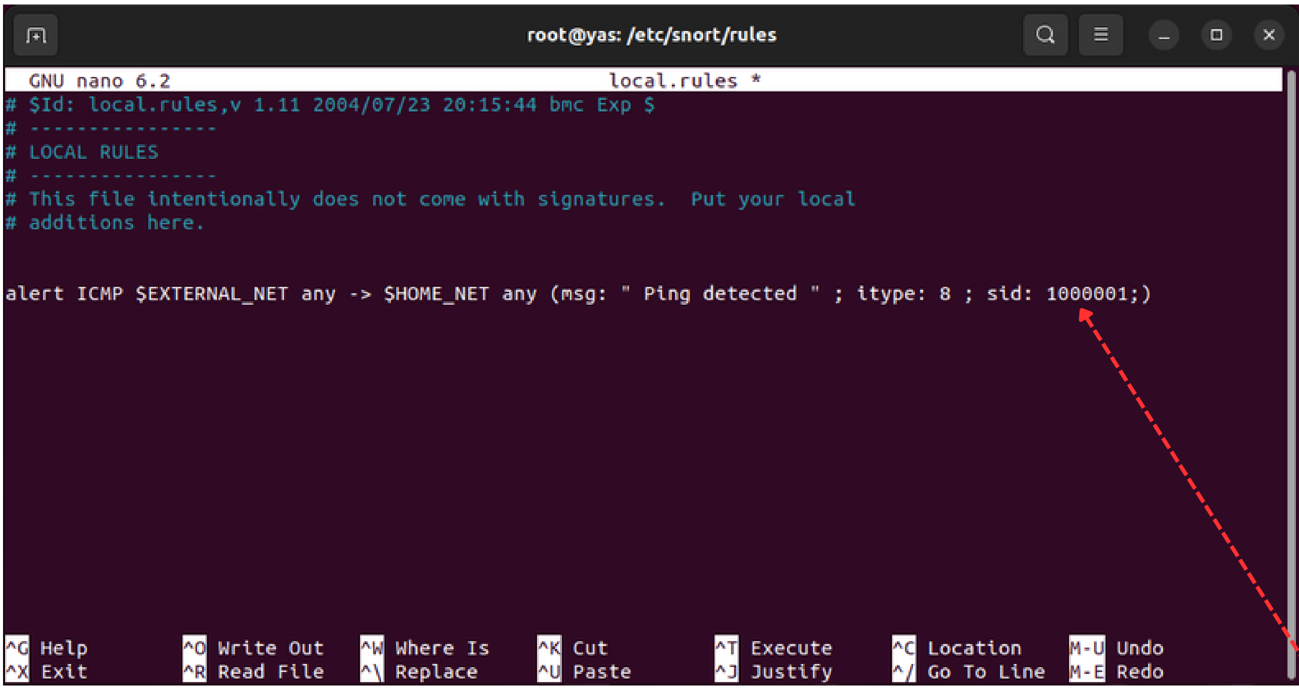
- ## Ping detection :

We can write a rule to detect a simple ping from an host on the EXTERNAL_NET directed to an host inside the HOME_NET.
To do so we can add our specific rules on the local.rules file .



```
root@yas: /etc/snort
GNU nano 6.2                         snort.conf
#
# If you install the official VRT Sourcefire rules please review this
# configuration file and re-enable (remove the comment in the first line) those
# rules files that are available in your system (in the /etc/snort/rules
# directory)

# site specific rules
include $RULE_PATH/local.rules

# The include files commented below have been disabled
# because they are not available in the stock Debian
# rules. If you install the Sourcefire VRT please make
# sure you re-enable them again:

#include $RULE_PATH/app-detect.rules
include $RULE_PATH/attack-responses.rules
include $RULE_PATH/backdoor.rules
include $RULE_PATH/bad-traffic.rules
#include $RULE_PATH/blacklist.rules
#include $RULE_PATH/botnet-cnc.rules

^G Help        ^O Write Out  ^W Where Is   ^K Cut        ^T Execute    ^C Location
^X Exit        ^R Read File  ^\ Replace    ^U Paste      ^J Justify    ^/ Go To Line
```

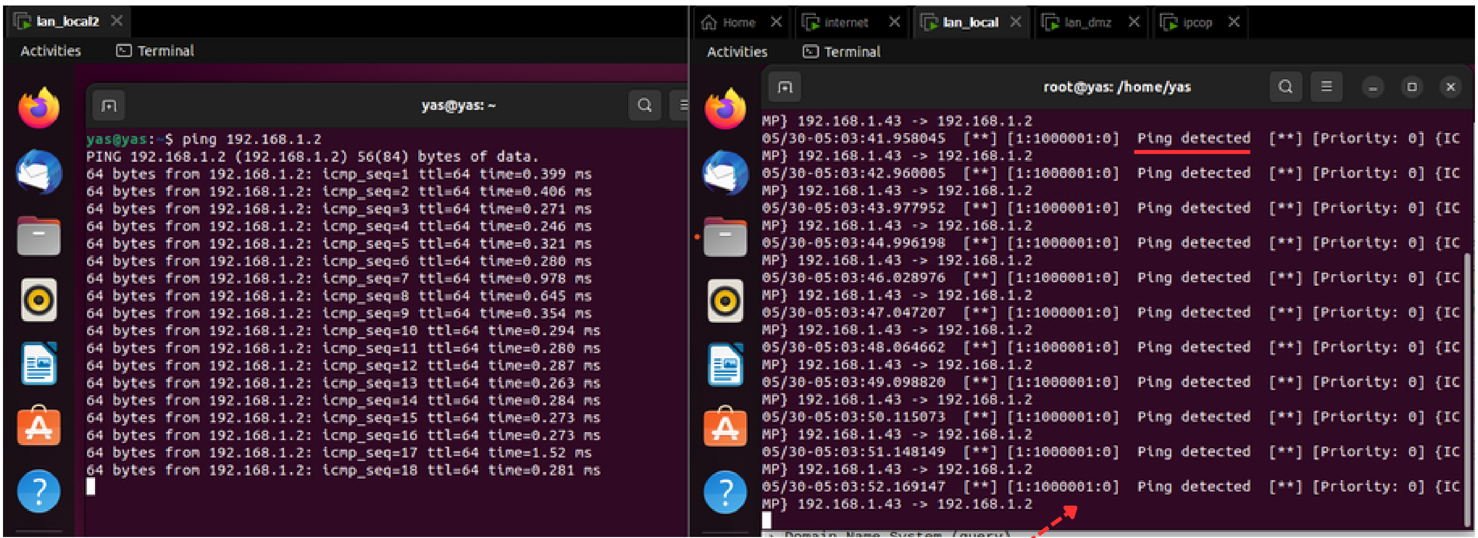
First we need to uncomment the local.rules file path on the snort.conf file.



This rule defines that an alert will be logged if an ICMP packet from the external network ($EXTERNAL_NET) and targeting the internal network ($HOME_NET).

The signature ID(sid) should be greater than 1000000 for our own rules,

we should restart snort first before the simulation .



Iwhen a ping is sent from the attacker using the command: ping 192.168.1.2 an alert displayed by Snort on the terminal.

• Detecting a SYN flood attack :

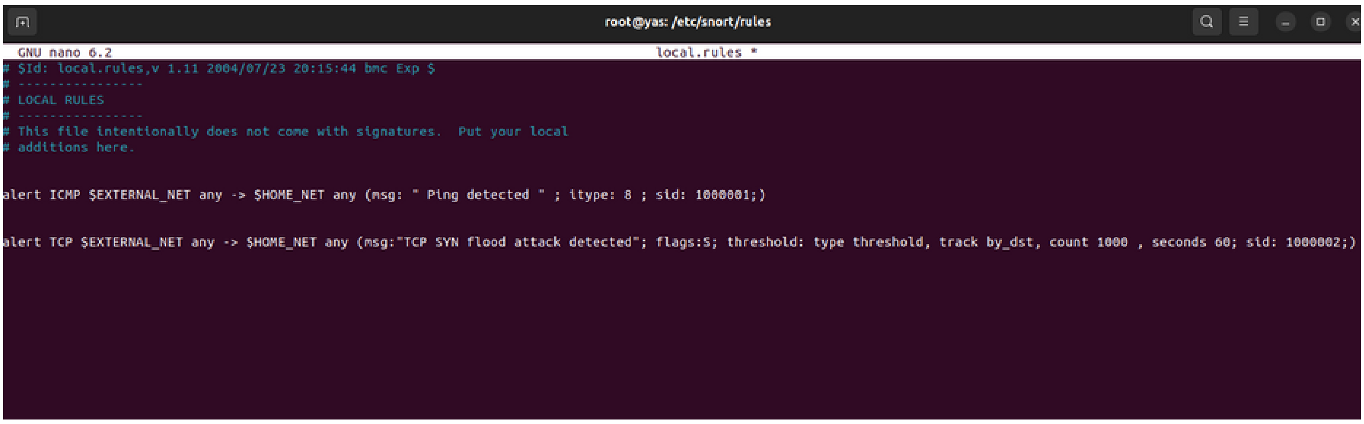
A SYN flood attack is a kind of denial-of-service attack in which the attacker sends a succession of SYN requests to a target's system in order to consume enough server resources to make the system unresponsive to legitimate traffic.

Implementation :

In order to simulate this attack a python script has been used to perform the SYN flood attack. The packet generator Scapy is used to create the required ACK packets to be sent to port 80 from random IPs in order to perform again a normal attack to an http web server.

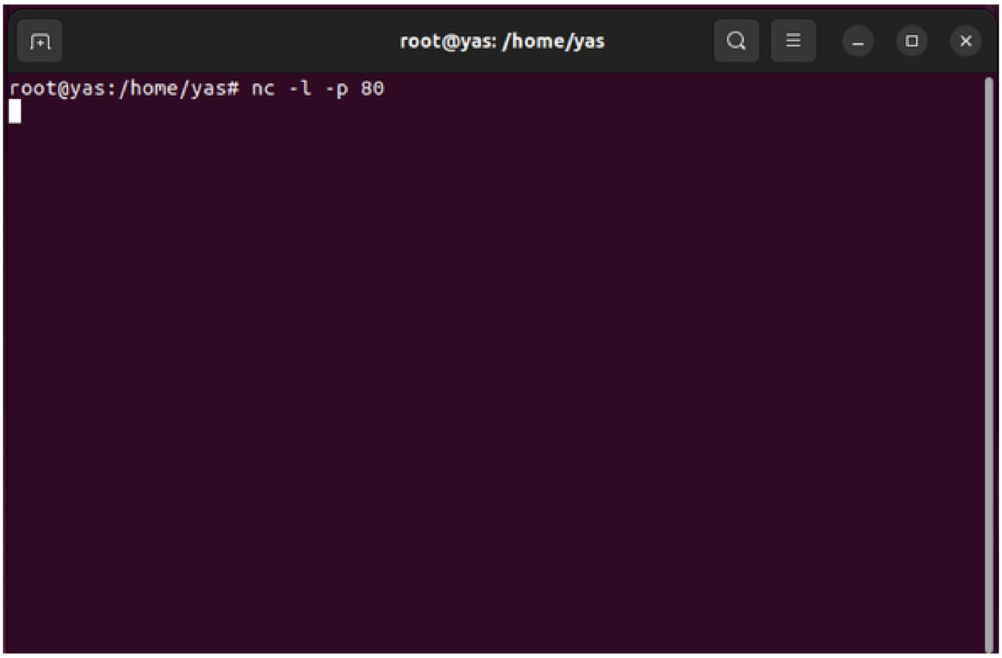
To detect a SYN flood attack whose target is the server inside the HOME_NET, a solution can be to track all the SYN packets with the same destination IP and if the receiving rate exceeds a predefined threshold an alert should be risen by Snort.

First we have to add the rule on the local.rules file.

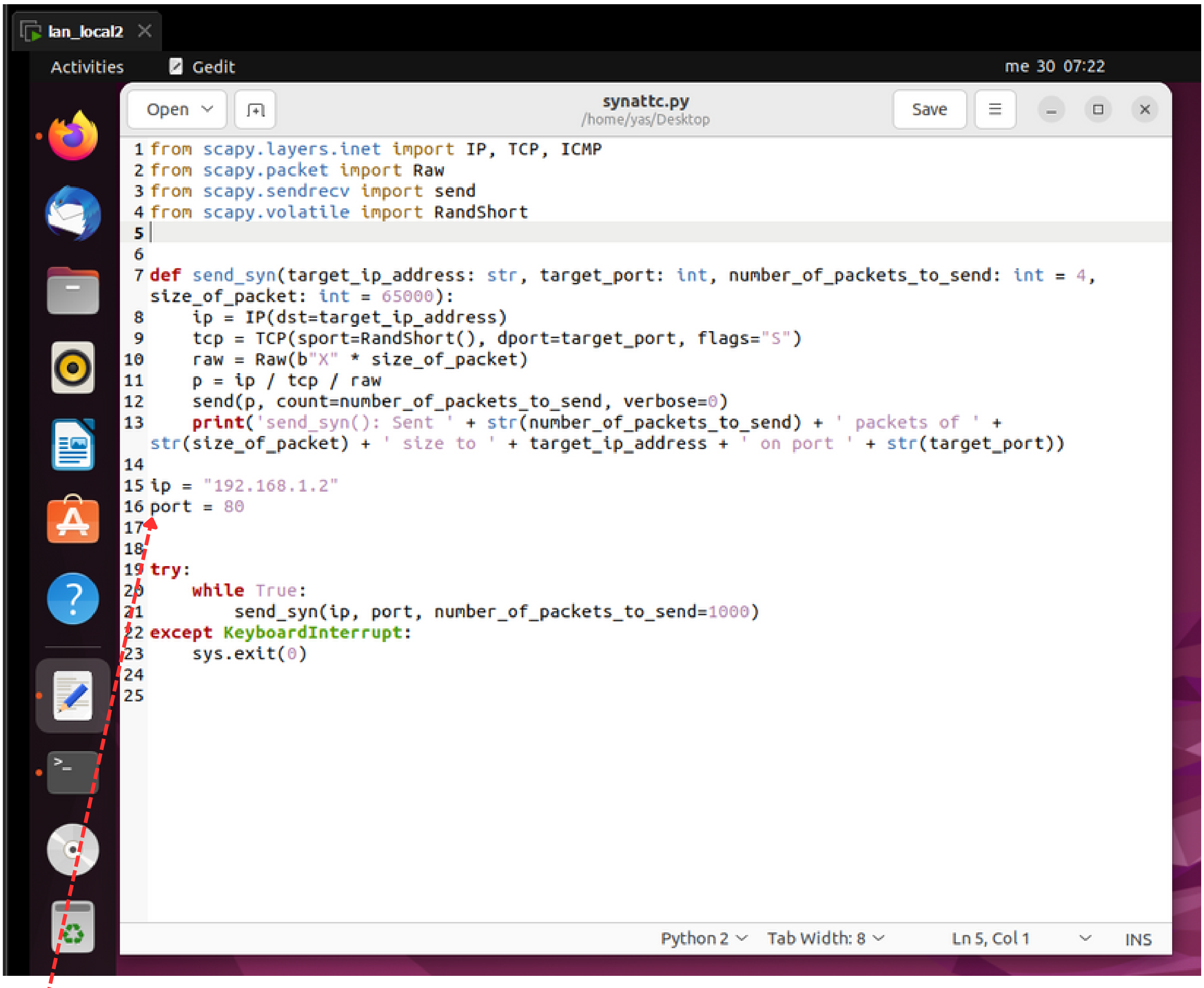


The rule detects TCP packets with the SYN flag set and generates an alert when a high count of SYN packets (1000 and more) is received at the destination IP within a 60-second window, indicating a possible TCP SYN flood attack.

Since the victim machine has to simulate a server ready to receive connection, we have to install Netcat. and make it listen for the TCP connection on port 80:

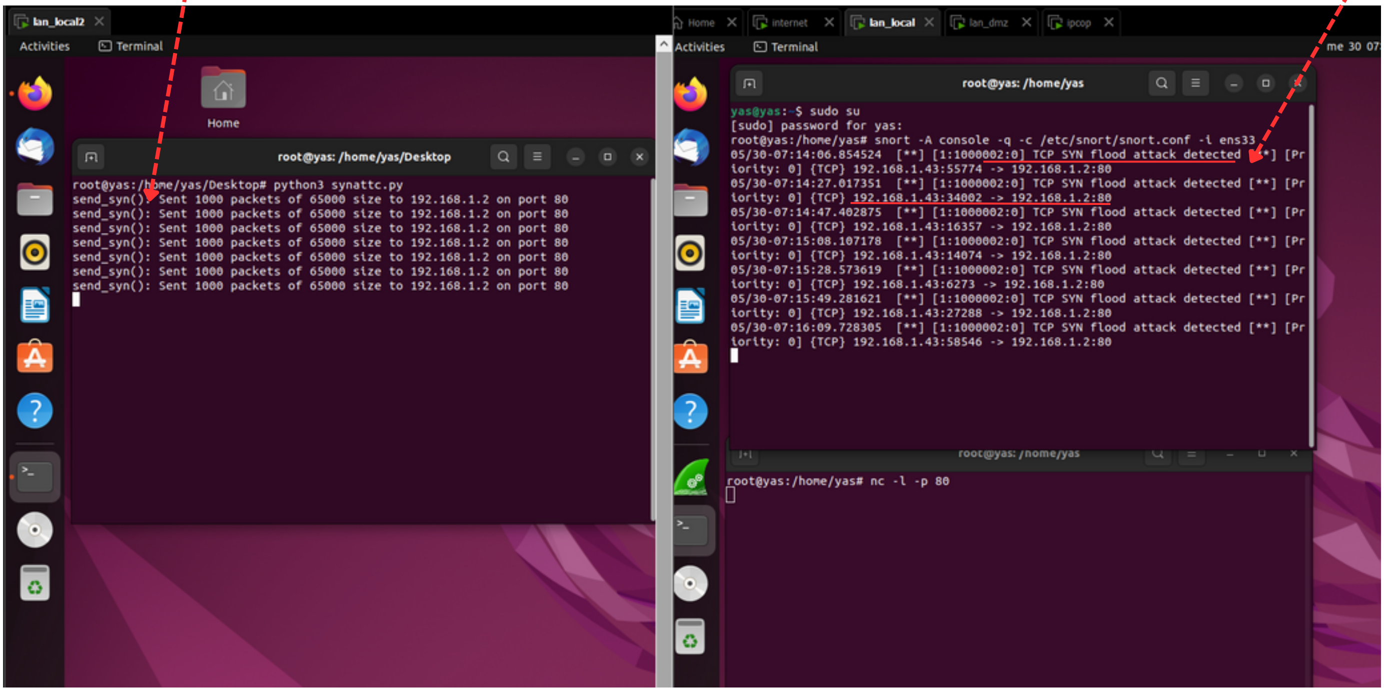


On the attacker machine a script to perform the attack has been created.

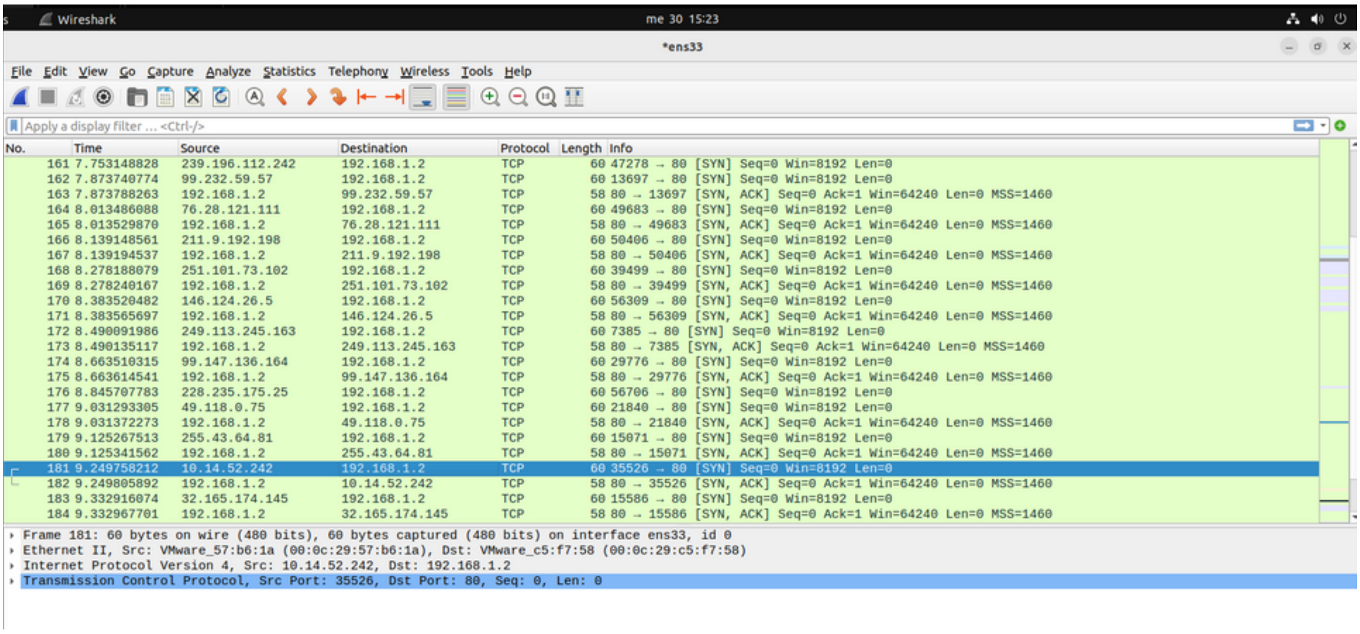


IP address of the target machine , and the specific port 80 how many packets has been sent up to now

the alert “TCP SYN flood attack detected” should be raised every 1000 sent packets

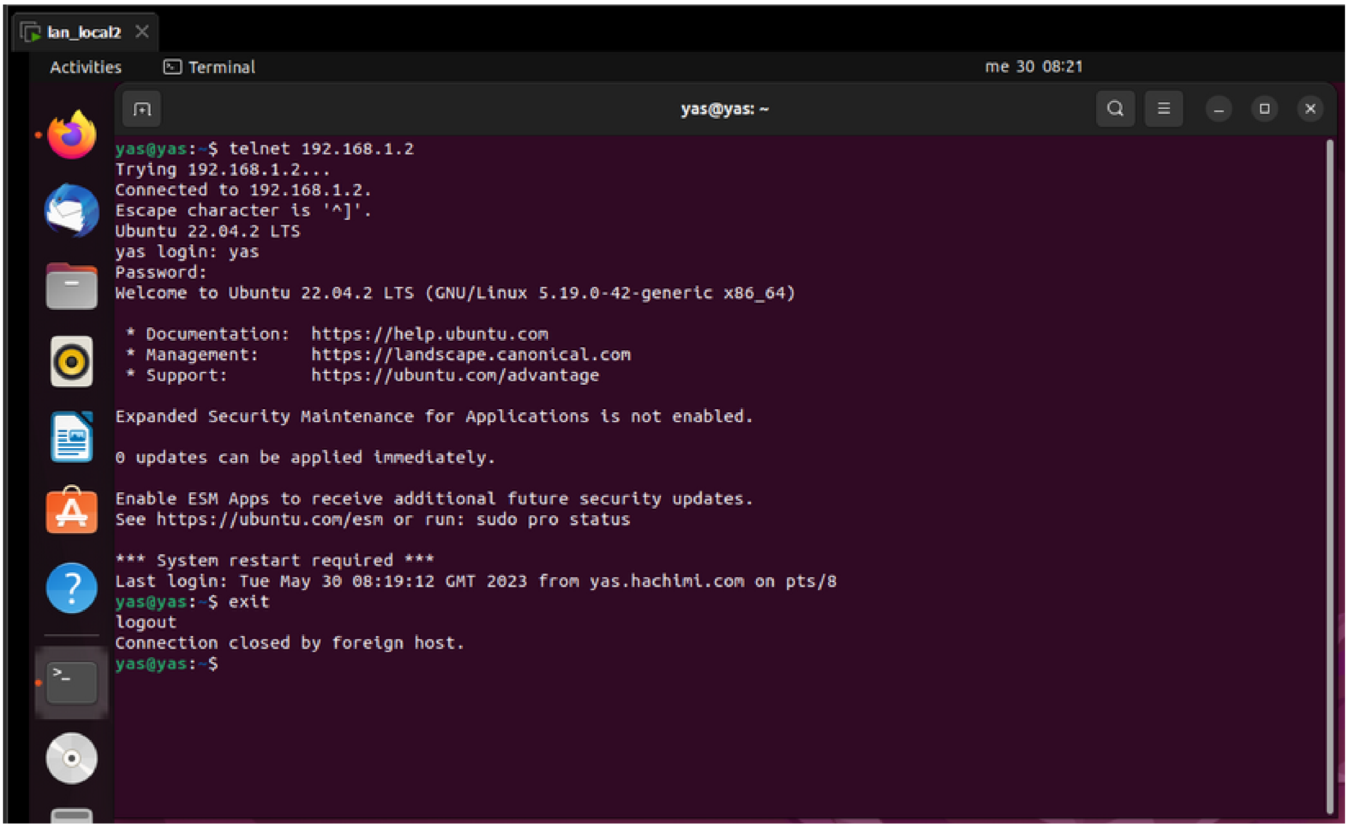


On the victim side Wireshark shows that SYN packets alternated with SYN ACK sent by the victim to spoofed IP that never respond and a lot of retransmission of the SYN ACK because there are no ACK answer to them.

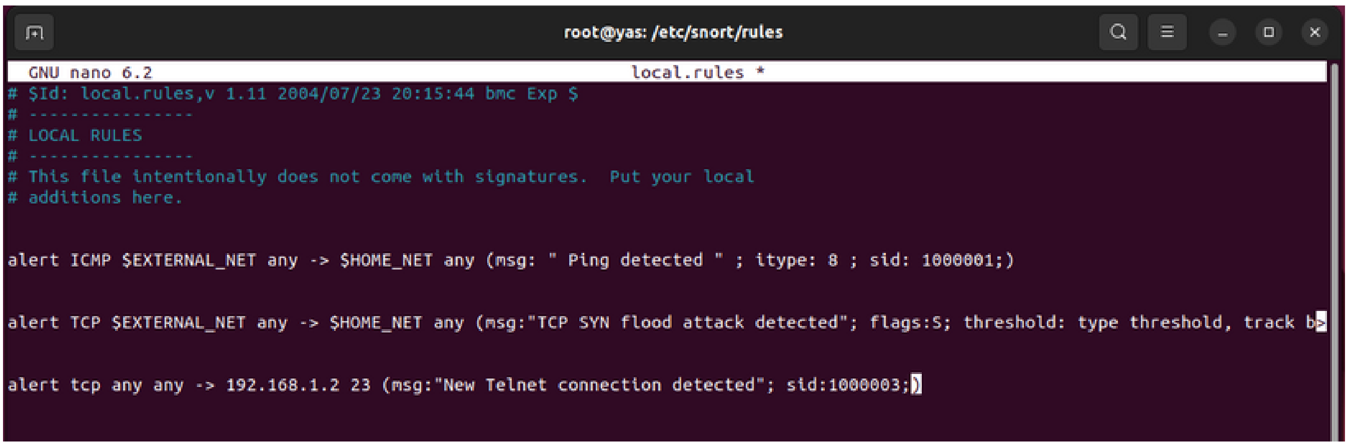


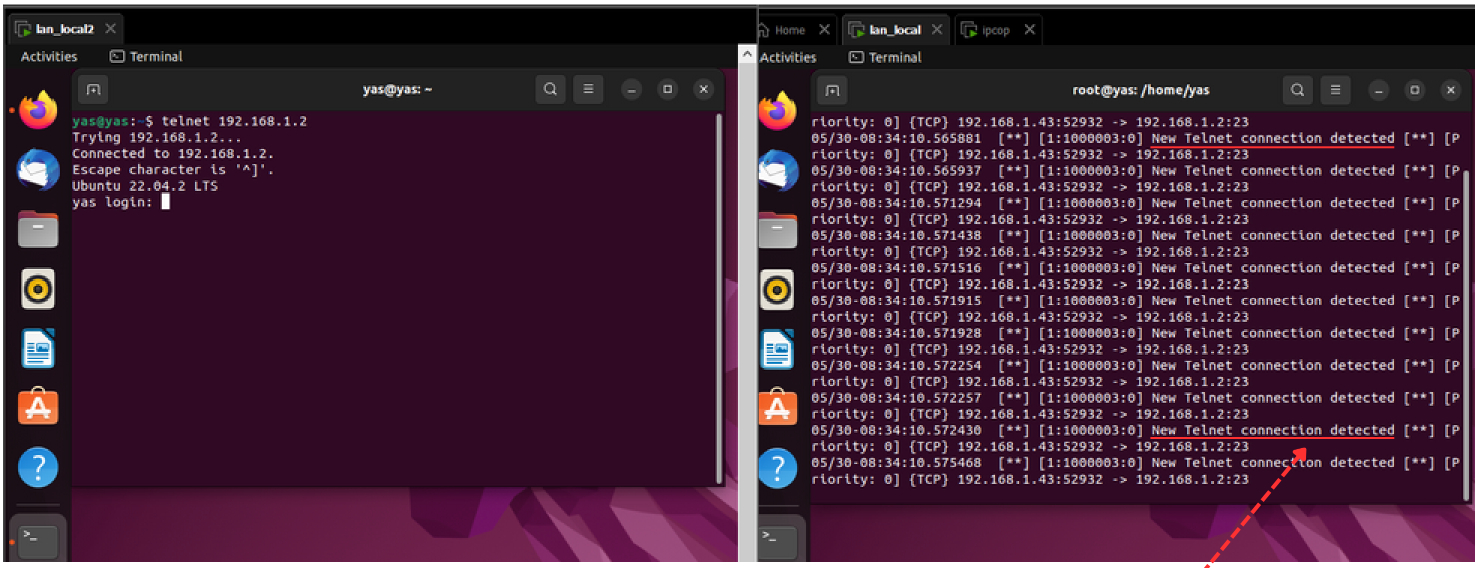
In conclusion now we are able to monitor by snort if a SYN flood attack is passing inside our network and it allows us to make a informed prevention using a firewall or Snort running in IPS mode.

- # Test Telnet from Attacker :



Now add a rule so that Snort will generate an alert with the message <new telnet connection, if the attacker tries to Telnet to "Telnet-Server" through port 23.





Snort generate an alert , whenever it detects incoming Telnet traffic to the specified destination IP address on port 23.

Conclusion :

In conclusion, the Snort lab conducted in this study aimed to explore the capabilities of Snort as a network intrusion detection system (NIDS) in a local network environment. The lab utilized two machines within the same LAN segment, with one machine acting as an attacker and the other as a victim.

Throughout the lab, various network attacks and scanning techniques were simulated to evaluate Snort's effectiveness in detecting and alerting on malicious activities. The lab primarily focused on internet attacks due to their potential severity and impact on network security.

By deploying Snort on the victim machine, it functioned as a NIDS, continuously monitoring network traffic and analyzing packets for known attack signatures and patterns. The default rule set provided by Snort covered a wide range of network threats, while custom rules were also introduced to enhance detection capabilities specific to the lab's network environment.

Overall, this Snort lab provided hands-on experience in deploying and configuring a NIDS, evaluating its effectiveness in detecting and alerting on network attacks. The insights gained from this lab can be applied to real-world network security scenarios, helping organizations strengthen their overall network defenses and better protect against evolving threats