INTERNSHIP ORGANIZED BY CYBER SECURITY CLUB

Project Report on

Setting up an Intrusion Detection System (IDS)

and Intrusion Prevention System (IPS)

BACHELOR OF ENGINEERING

IN

COMPUTER ENGINEERING

BY

Saitwadekar Valay Angar (Roll No. 49)

TE CMPN

DEPARTMENT OF COMPUTER ENGINEERING

CYBER SECURITY CLUB

(SWDC)

Accredited by NBA for 3 years w.e.f. 1st July 2022

SHREE L. R. TIWARI COLLEGE OF ENGINEERING

SHREE L.R. TIWARI EDUCATIONAL CAMPUS, MIRA ROAD (East),

THANE - 401 107, MAHARASHTRA.

University of Mumbai

(AY 2022-23)

Table of Contents

Introduction

Intrusion detection is the process of monitoring your network traffic and analyzing it for signs of possible intrusions, such as exploit attempts and incidents that may be imminent threats to your network. For its part, intrusion prevention is the process of performing intrusion detection and then stopping the detected incidents, typically done by dropping packets or terminating sessions. These security measures are available as intrusion detection systems (IDS) and intrusion prevention systems (IPS), which are part of network security measures taken to detect and stop potential incidents and are included functionality within next-generation firewalls (NGFW).

IDS/IPS monitors all traffic on the network to identify any known malicious behavior. One of the ways in which an attacker will try to compromise a network is by exploiting a vulnerability within a device or within software. IDS/IPS identifies those exploit attempts and blocks them before they successfully compromise any endpoints within the network. IDS/IPS are necessary security technologies, both at the network edge and within the data center, precisely because they can stop attackers while they are gathering information about your network.

Both systems can:

Monitor. After setup, these programs can look over traffic within parameters you specify, and they will work until you turn them off.

Alert. Both programs will send a notification to those you specify when a problem has been spotted.

Learn. Both can use machine learning to understand patterns and emerging threats.

Log. Both will keep records of attacks and responses, so you can adjust your protections accordingly.

But they differ due to:

Response. An IDS is passive, while an IPS is an active control system. You must take action after an IDS alerts you, as your system is still under attack.

Protection. Arguably, an IDS offers less help when you're under threat. You must figure out what to do, when to do it, and how to clean up the mess. An IPS does all of this for you.

False positives. If an IDS gives you an alert about something that isn't troublesome at all, you're the only one inconvenienced. If an IPS shuts down traffic, many people could be impacted.

Hardware/Software Details

An IDS/IPS is usually run on a Dedicated Server type of Hardware with Networking Capabilites. It uses special Propreitary Software on a Linux/Unix based Operating System.

In my case, I have used Kali Linux running on a local machine along with SNORT which is an Open Source IPS and IDS Solution.

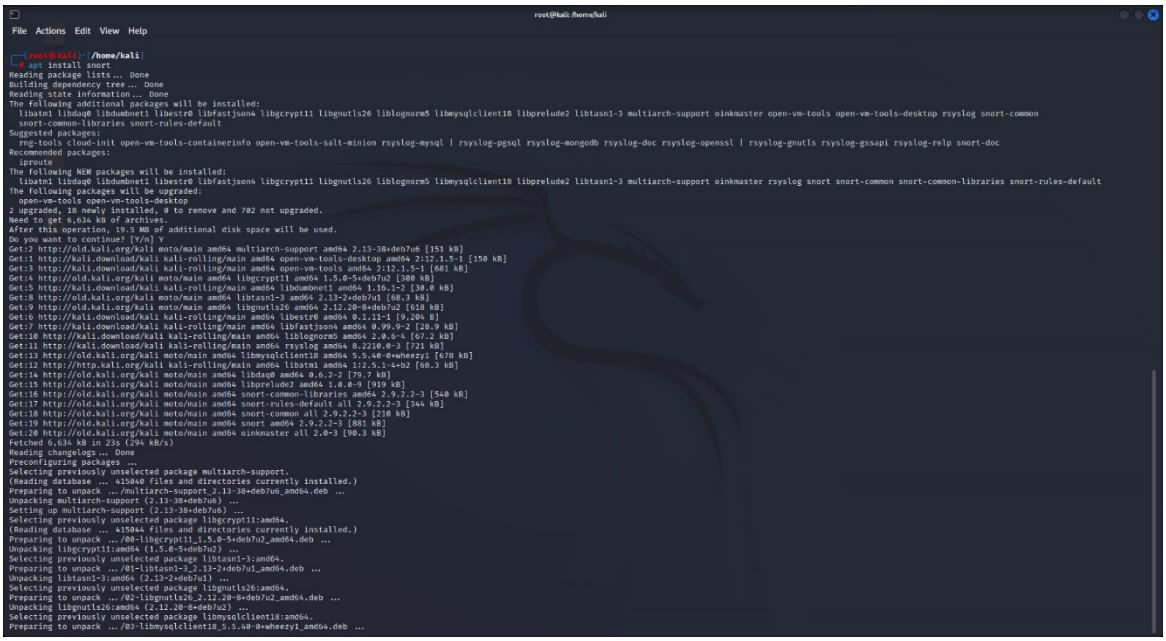
Procedure/Steps

First we Install the Snort Packages and Update our Kali Linux Dependencies.

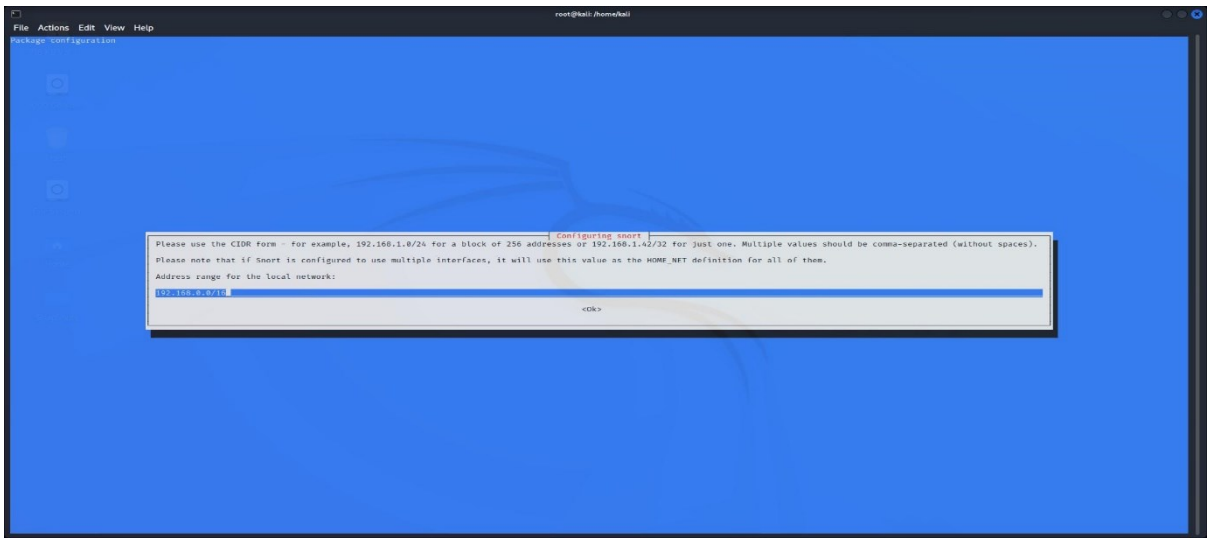
We are in Super User mode so we avoid all the sudo commands.

To Setup Snort as IDS:

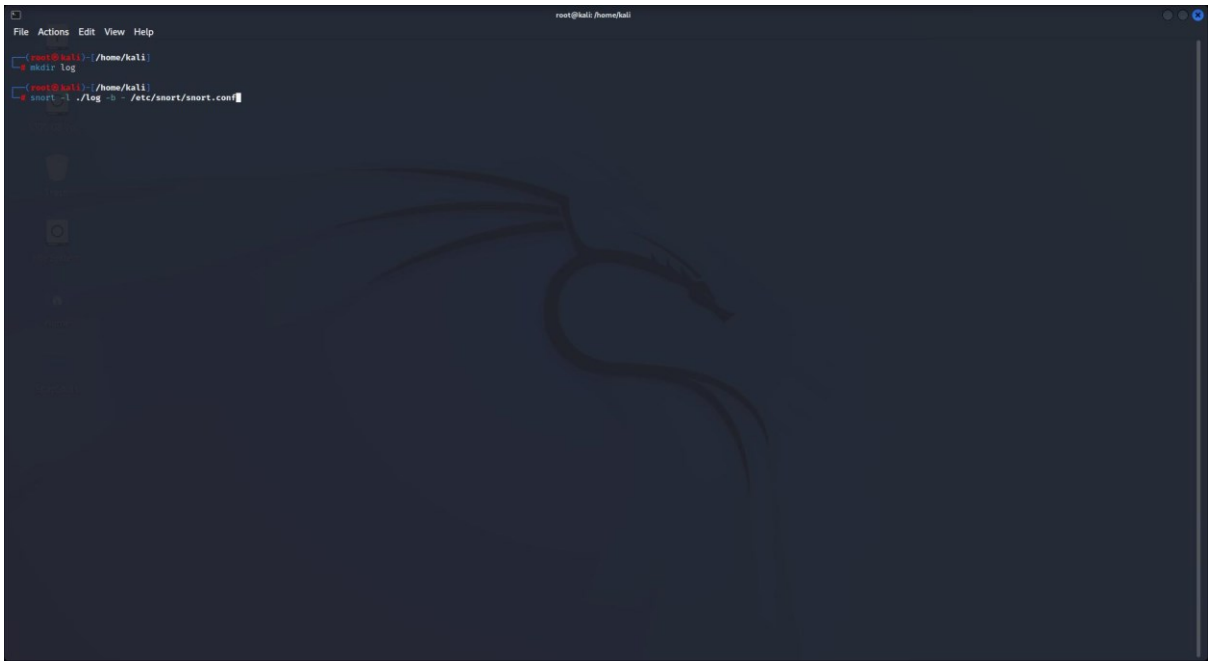
1) apt install snort



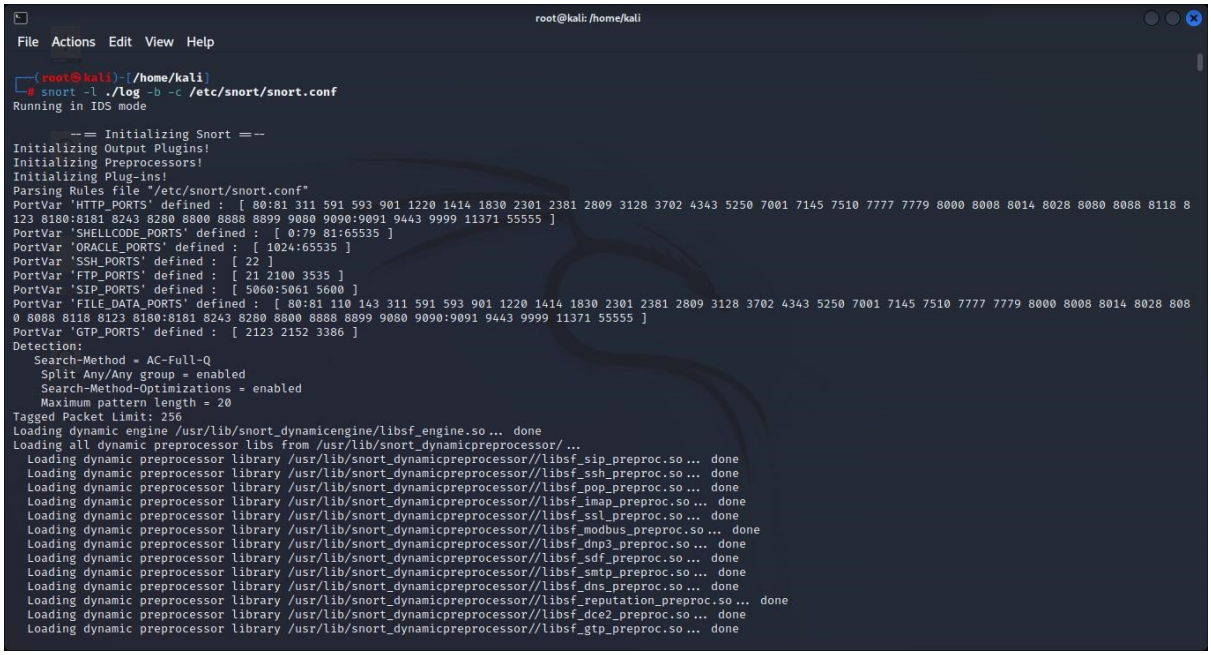
It asks for Address Range of Local Network while performing Configuration



2) Create a Folder for storing Logs and Create a new Log for the Current Session



3) Run Snort to check if it is working



```
root@kali: /home/kali/log
File  Actions  Edit  View  Help
SMTP Preprocessor Statistics
  Total sessions                                   : 0
  Max concurrent sessions                          : 0
dcerpc2 Preprocessor Statistics
  Total sessions: 0
SSL Preprocessor:
   SSL packets decoded: 12
           Client Hello: 2
           Server Hello: 2
            Certificate: 0
            Server Done: 0
    Client Key Exchange: 0
    Server Key Exchange: 0
         Change Cipher: 3
              Finished: 0
     Client Application: 4
     Server Application: 4
                  Alert: 2
  Unrecognized records: 0
  Completed handshakes: 0
        Bad handshakes: 0
      Sessions ignored: 3
    Detection disabled: 0
SIP Preprocessor Statistics
  Total sessions: 0
Snort exiting

┌──(root㉿kali)-[/home/kali]
└─# cd log

┌──(root㉿kali)-[/home/kali/log]
└─# ls
alert  snort.log.1672518527

┌──(root㉿kali)-[/home/kali/log]
└─#
```

Checking for current Logs

4) Open the Configuration File

```
root@kali: /home/kali/log
File  Actions  Edit  View  Help
  GNU nano 6.4                                    /etc/snort/snort.conf
#--------------------------------------------------
#   VRT Rule Packages Snort.conf
#
#   For more information visit us at:
#     http://www.snort.org                    Snort Website
#     http://vrt-sourcefire.blogspot.com/     Sourcefire VRT Blog
#
#     Mailing list Contact:      snort-sigs@lists.sourceforge.net
#     False Positive reports:    fp@sourcefire.com
#     Snort bugs:                bugs@snort.org
#
#     Compatible with Snort Versions:
#     VERSIONS : 2.9.2.2
#
#     Snort build options:
#     OPTIONS : --enable-ipv6 --enable-gre --enable-mpls --enable-targetbased --enable-decoder-preprocessor-rules --enable-ppm --enable-perfprofiling --enable-zlib --
#
#     Additional information:
#     This configuration file enables active response, to run snort in
#     test mode -T you are required to supply an interface -i <interface>
#     or test mode will fail to fully validate the configuration and
#     exit with a FATAL error
#--------------------------------------------------

###################################################
# This file contains a sample snort configuration.
# You should take the following steps to create your own custom configuration:
#
#  1) Set the network variables.
#  2) Configure the decoder
#  3) Configure the base detection engine
#  4) Configure dynamic loaded libraries
#  5) Configure preprocessors
#  6) Configure output plugins
#  7) Customize your rule set
#  8) Customize preprocessor and decoder rule set
#  9) Customize shared object rule set
                                          [ Read 680 lines ]
^G Help      ^O Write Out  ^W Where Is   ^K Cut      ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket  M-Q Previous
^X Exit      ^R Read File  ^\ Replace    ^U Paste    ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was    M-W Next
```

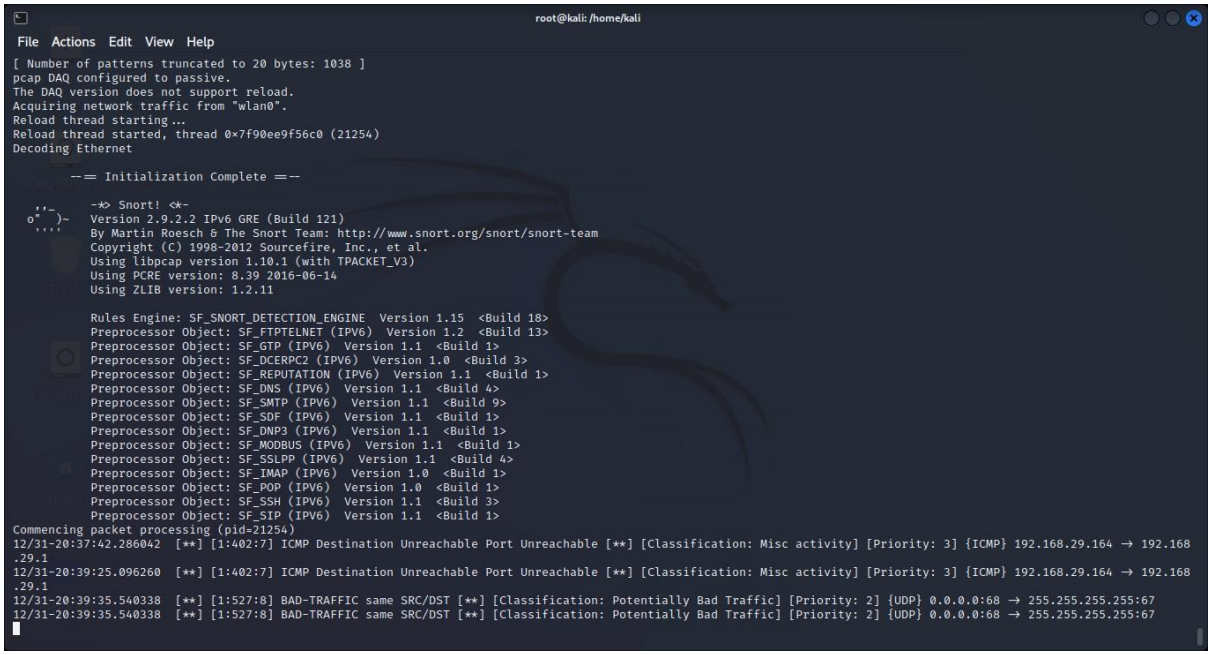
nano etc/snort/snort.conf

```
root@kali: /home/kali/log
File  Actions  Edit  View  Help
  GNU nano 6.4                                   /etc/snort/snort.conf *

# Setup the network addresses you are protecting
ipvar HOME_NET 192.168.0.0/16

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
#ipvar EXTERNAL_NET !$HOME_NET

# List of DNS servers on your network
ipvar DNS_SERVERS $HOME_NET

# List of SMTP servers on your network
ipvar SMTP_SERVERS $HOME_NET

# List of web servers on your network
ipvar HTTP_SERVERS $HOME_NET

# List of sql servers on your network
ipvar SQL_SERVERS $HOME_NET

# List of telnet servers on your network
ipvar TELNET_SERVERS $HOME_NET

# List of ssh servers on your network
ipvar SSH_SERVERS $HOME_NET

# List of ftp servers on your network
ipvar FTP_SERVERS $HOME_NET

# List of sip servers on your network
ipvar SIP_SERVERS $HOME_NET

# List of ports you run web servers on
portvar HTTP_PORTS [80,81,311,591,593,901,1220,1414,1830,2301,2381,2809,3128,3702,4343,5250,7001,7145,7510,7777,7779,8000,8008,8014,8028,8080,8088,8118,8123,8180,8181>

# List of ports you want to look for SHELLCODE on.
portvar SHELLCODE_PORTS !80

^G Help      ^O Write Out  ^W Where Is   ^K Cut       ^T Execute    ^C Location   M-U Undo      M-A Set Mark  M-] To Bracket  M-Q Previous
^X Exit      ^R Read File  ^\ Replace    ^U Paste     ^J Justify    ^/ Go To Line M-E Redo      M-6 Copy      ^Q Where Was    M-W Next
```

Update the IP Address to be Monitored as Home

5) Run Snort again to check if it monitors the Incoming Traffic

```
root@kali: /home/kali
File  Actions  Edit  View  Help

┌──(root㉿kali)-[/home/kali]
└─# sudo snort -d -l /var/log/snort -h 192.168.0.016 -A console -c /etc/snort/snort.conf
Running in IDS mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/snort.conf"
PortVar 'HTTP_PORTS' defined :  [ 80:81 311 591 593 901 1220 1414 1830 2301 2381 2809 3128 3702 4343 5250 7001 7145 7510 7777 7779 8000 8008 8014 8028 8080 8088 8118 8
123 8180:8181 8243 8280 8800 8888 8899 9080 9090:9091 9443 9999 11371 55555 ]
PortVar 'SHELLCODE_PORTS' defined :  [ 0:79 81:65535 ]
PortVar 'ORACLE_PORTS' defined :  [ 1024:65535 ]
PortVar 'SSH_PORTS' defined :  [ 22 ]
PortVar 'FTP_PORTS' defined :  [ 21 2100 3535 ]
PortVar 'SIP_PORTS' defined :  [ 5060:5061 5600 ]
PortVar 'FILE_DATA_PORTS' defined :  [ 80:81 110 143 311 591 593 901 1220 1414 1830 2301 2381 2809 3128 3702 4343 5250 7001 7145 7510 7777 7779 8000 8008 8014 8028 808
0 8088 8118 8123 8180:8181 8243 8280 8800 8888 8899 9080 9090:9091 9443 9999 11371 55555 ]
PortVar 'GTP_PORTS' defined :  [ 2123 2152 3386 ]
Detection:
   Search-Method = AC-Full-Q
    Split Any/Any group = enabled
    Search-Method-Optimizations = enabled
    Maximum pattern length = 20
Tagged Packet Limit: 256
Loading dynamic engine /usr/lib/snort_dynamicengine/libsf_engine.so ...  done
Loading all dynamic preprocessor libs from /usr/lib/snort_dynamicpreprocessor/ ...
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_sip_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_ssh_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_pop_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_imap_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_ssl_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_modbus_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_dnp3_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_sdf_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_smtp_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_dns_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_reputation_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_dce2_preproc.so ...  done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_gtp_preproc.so ...  done
```

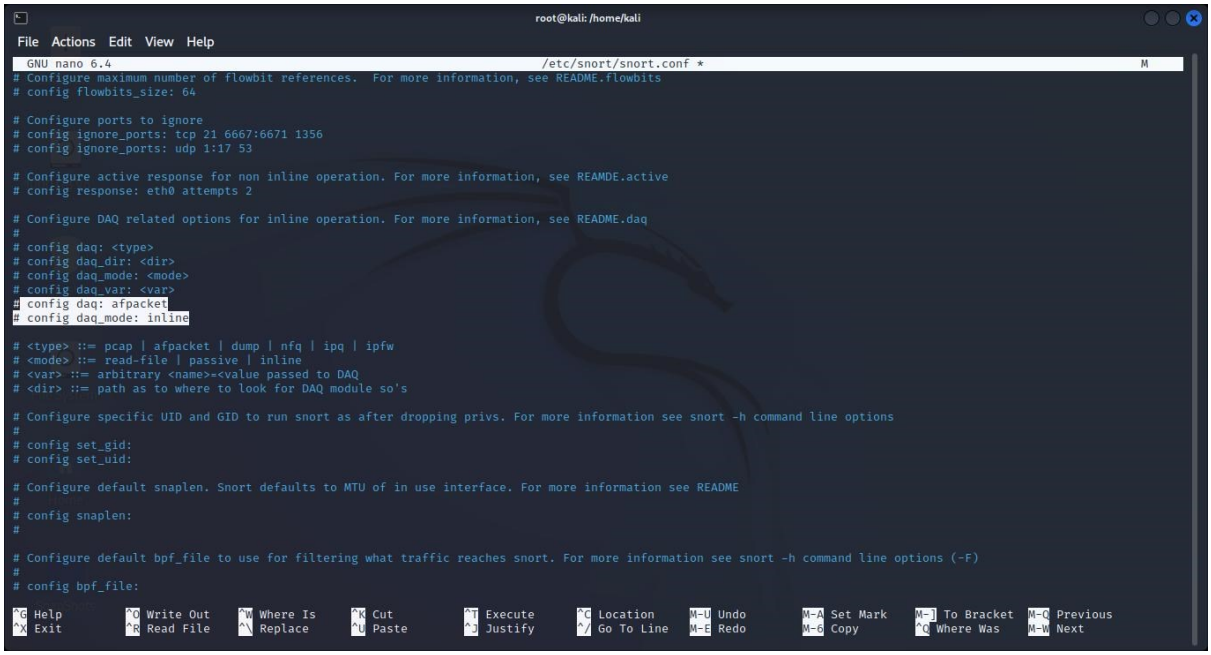


Snort is Monitoring the Traffic and showing Logs

IDS is Setup

To setup Snort as both IPS and IDS:

1) Configure Snort to run in “inline” mode and in afpacket daq



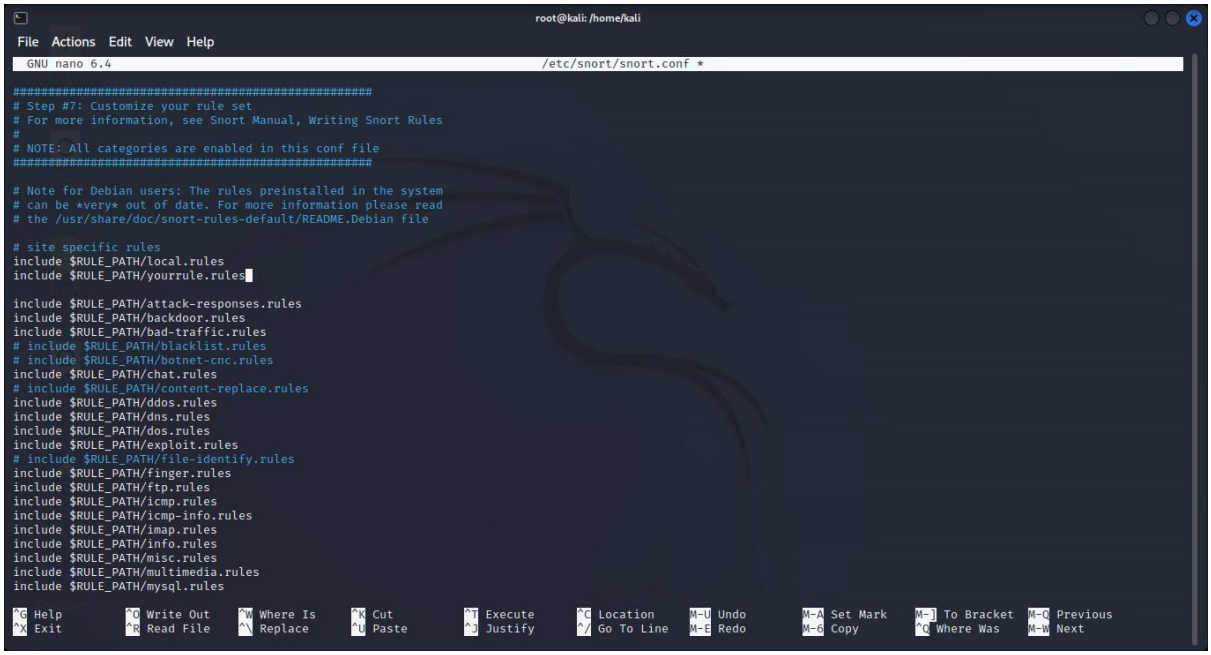
2) Run Snort in Test Mode to check if it works in “inline”

```
root@kali: /home/kali
File  Actions  Edit  View  Help

┌──(root㉿kali)-[/home/kali]
└─# sudo snort -T -c /etc/snort/snort.conf -Q -i eth1:eth2
Enabling inline operation
Running in Test mode

        --== Initializing Snort ==--
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plug-ins!
Parsing Rules file "/etc/snort/snort.conf"
PortVar 'HTTP_PORTS' defined :  [ 80:81 311 591 593 901 1220 1414 1830 2301 2381 2809 3128 3702 4343 5250 7001 7145 7510 7777 7779 8000 8008 8014 8028 8080 8088 8118
8123 8180:8181 8243 8280 8800 8888 8899 9080 9090:9091 9443 9999 11371 55555 ]
PortVar 'SHELLCODE_PORTS' defined :  [ 0:79 81:65535 ]
PortVar 'ORACLE_PORTS' defined :  [ 1024:65535 ]
PortVar 'SSH_PORTS' defined :  [ 22 ]
PortVar 'FTP_PORTS' defined :  [ 21 2100 3535 ]
PortVar 'SIP_PORTS' defined :  [ 5060:5061 5600 ]
PortVar 'FILE_DATA_PORTS' defined :  [ 80:81 110 143 311 591 593 901 1220 1414 1830 2301 2381 2809 3128 3702 4343 5250 7001 7145 7510 7777 7779 8000 8008 8014 8028 80
80 8088 8118 8123 8180:8181 8243 8280 8800 8888 8899 9080 9090:9091 9443 9999 11371 55555 ]
PortVar 'GTP_PORTS' defined :  [ 2123 2152 3386 ]
Detection:
   Search-Method = AC-Full-Q
    Split Any/Any group = enabled
    Search-Method-Optimizations = enabled
    Maximum pattern length = 20
Tagged Packet Limit: 256
Loading dynamic engine /usr/lib/snort_dynamicengine/libsf_engine.so ... done
Loading all dynamic preprocessor libs from /usr/lib/snort_dynamicpreprocessor/ ...
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_sip_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_ssh_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_pop_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_imap_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_ssl_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_modbus_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_dnp3_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_sdf_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_smtp_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_dns_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_reputation_preproc.so ... done
  Loading dynamic preprocessor library /usr/lib/snort_dynamicpreprocessor//libsf_dce2_preproc.so ... done
```

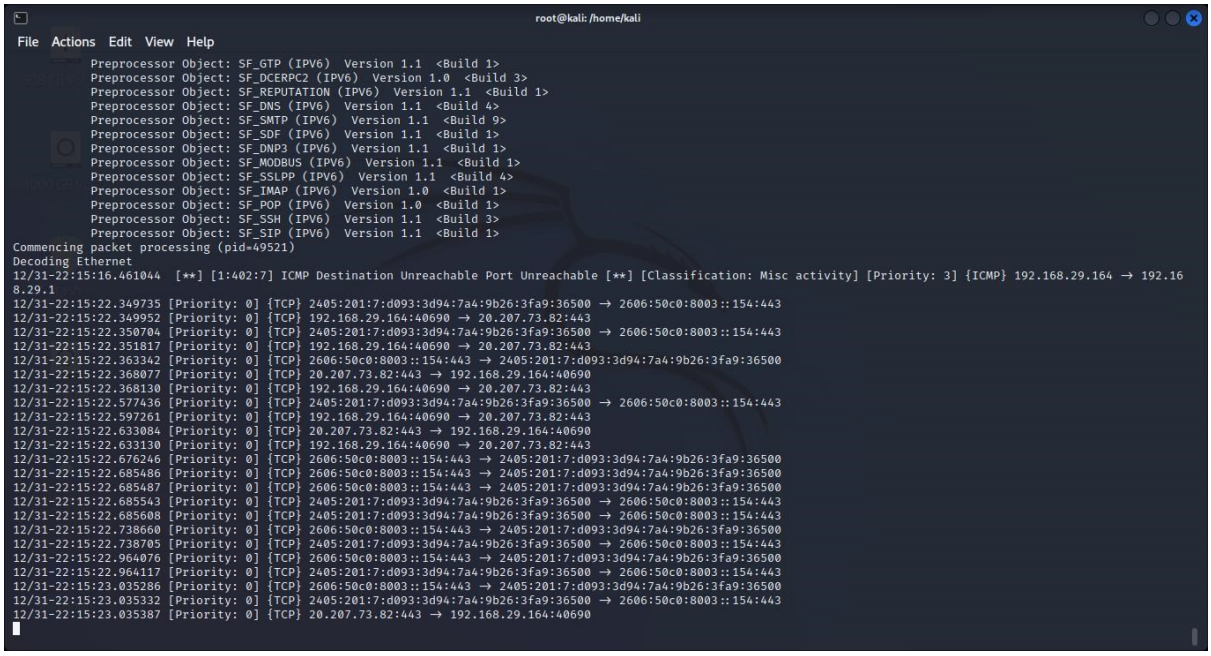
3) Add Rule to drop ICMP and TCP Packets

```
root@kali: /home/kali
File  Actions  Edit  View  Help
  GNU nano 6.4                                  /etc/snort/rules/yourrule.rules *
alert icmp any any -> $HOME_NET any (msg:"ICMP test detected"; GID:1; sid:10000001; rev:001; classtype:icmp-event;)

File Name to Write: /etc/snort/rules/yourrule.rules
^G Help          M-D DOS Format    M-A Append        M-B Backup File
^C Cancel        M-M Mac Format    M-P Prepend       ^T Browse
```

4) Add the rules file to the main conf file



5) Save the file and run Snort to work as both IPS and IDS.

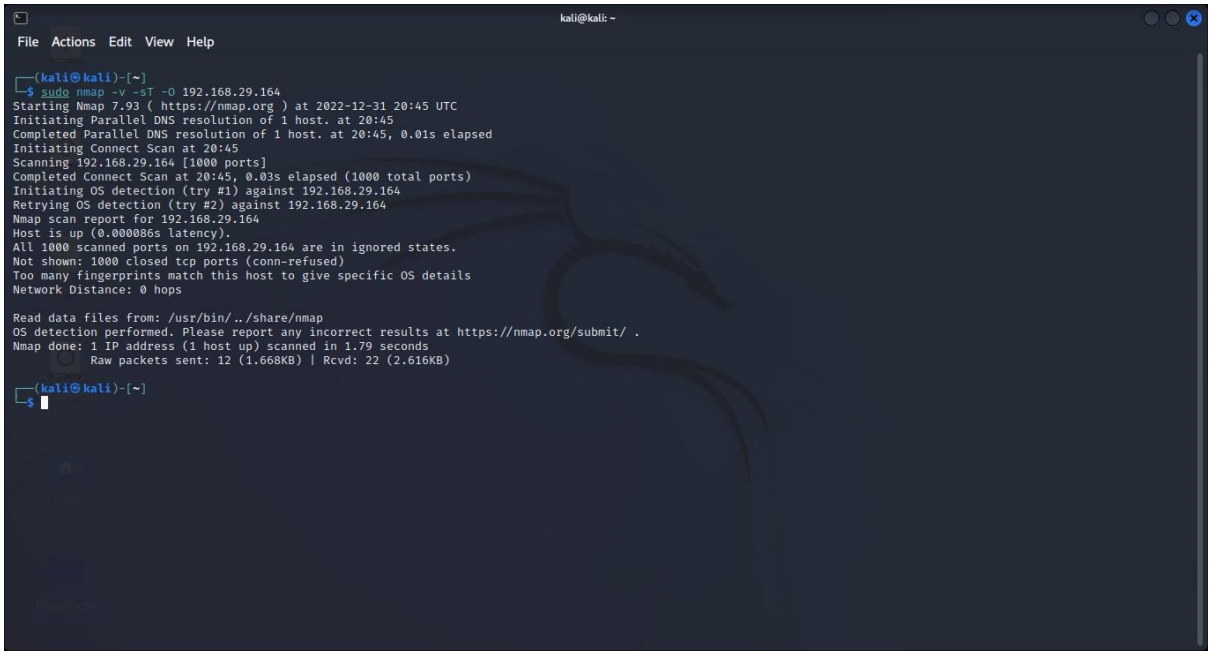


Snort is Monitoring the Traffic and showing Logs.

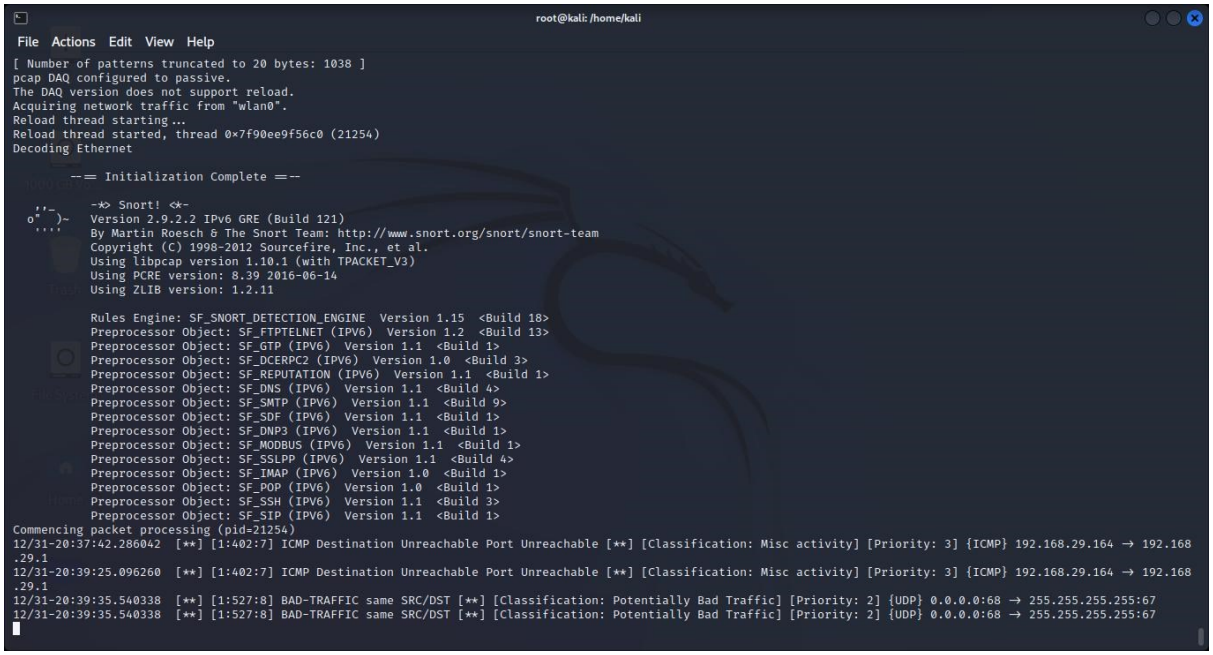
Snort is Setup as both IPS and IDS.

Output

Performing nmap from other device on the IDS that we setup



Snort Detecting and showing Bad Traffic in Logs



Performing 2 attacks to check if Snort drops the Packets

```
kali@kali: ~
File  Actions  Edit  View  Help

┌──(kali㉿kali)-[~]
└─$ sudo nmap -v -sT -O 192.168.29.164
Starting Nmap 7.93 ( https://nmap.org ) at 2022-12-31 21:54 UTC
Initiating Parallel DNS resolution of 1 host. at 21:54
Completed Parallel DNS resolution of 1 host. at 21:54, 0.00s elapsed
Initiating Connect Scan at 21:54
Scanning 192.168.29.164 [1000 ports]
Completed Connect Scan at 21:54, 0.03s elapsed (1000 total ports)
Initiating OS detection (try #1) against 192.168.29.164
Retrying OS detection (try #2) against 192.168.29.164
Nmap scan report for 192.168.29.164
Host is up (0.00012s latency).
All 1000 scanned ports on 192.168.29.164 are in ignored states.
Not shown: 1000 closed tcp ports (conn-refused)
Too many fingerprints match this host to give specific OS details
Network Distance: 0 hops

Read data files from: /usr/bin/../share/nmap
OS detection performed. Please report any incorrect results at https://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 1.81 seconds
           Raw packets sent: 12 (1.668KB) | Rcvd: 22 (2.616KB)

┌──(kali㉿kali)-[~]
└─$ sudo hping3 -c 10000 -d 120 -S -w 64 -p 21 --flood --rand-source 192.168.29.164
HPING 192.168.29.164 (wlan0 192.168.29.164): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown
```

Snort Drops all Packages and returns 0 packets

```
root@kali: /home/kali
File  Actions  Edit  View  Help
12/31-22:22:58.509019  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.64.200.23:19192 → 192.168.2
9.1:21
12/31-22:22:58.632925  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.243.163.157:19460 → 192.168
.29.1:21
12/31-22:22:58.719245  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.185.219.154:20238 → 192.168
.29.1:21
12/31-22:22:58.726613  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.53.13.157:20326 → 192.168.2
9.1:21
12/31-22:22:58.727254  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.79.158.226:20335 → 192.168.
29.1:21
12/31-22:22:58.730935  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.8.142.54:20381 → 192.168.29
.1:21
12/31-22:22:58.743365  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.216.17.7:20523 → 192.168.29
.1:21
12/31-22:22:58.752979  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.172.177.254:20720 → 192.168
.29.1:21
12/31-22:22:58.947257  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.15.92.123:21397 → 192.168.2
9.1:21
12/31-22:22:59.001997  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.13.44.45:21849 → 192.168.29
.1:21
12/31-22:22:59.022352  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.7.166.223:22461 → 192.168.2
9.1:21
12/31-22:22:59.023087  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.142.174.165:22504 → 192.168
.29.1:21
12/31-22:22:59.027651  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.152.45.222:22638 → 192.168.
29.1:21
12/31-22:22:59.039160  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.60.1.32:22882 → 192.168.29.
1:21
12/31-22:22:59.041071  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.4.139.231:22995 → 192.168.2
9.1:21
12/31-22:22:59.044007  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.189.180.145:23116 → 192.168
.29.1:21
12/31-22:22:59.044718  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.242.75.162:23146 → 192.168.
29.1:21
12/31-22:22:59.048397  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.242.162.199:23230 → 192.168
.29.1:21
12/31-22:22:59.052681  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.245.205.126:23377 → 192.168
.29.1:21
12/31-22:22:59.055429  [**] [1:528:5] BAD-TRAFFIC loopback traffic [**] [Classification: Potentially Bad Traffic] [Priority: 2] {TCP} 127.39.243.107:23409 → 192.168.
29.1:21
```

References

1. https://www.snort.org/
2. https://www.snort.org/#documents
3. https://serverfault.com/questions/tagged/snort
4. https://security.stackexchange.com/questions/tagged/snort
5. https://github.com/snort3/snort3
6. https://suricata.io/