

How to Use the Snort IDS/IPS Complete Practical Guide

By **Tamil S** - May 9, 2023



Snort is a widely used open-source Network Intrusion Detection System (NIDS) that can analyze network traffic and detect potential security threats.

It works by analyzing network traffic in real time and comparing it against a set of rules, which the user or administrator defines.

It can detect various attacks, such as port scans, buffer overflows, and malware infections. When Snort detects an attack, it can generate an alert, log the event, and even block the offending traffic.

The tool [can be deployed](#) on various operating systems, including Windows, Linux, and macOS. It can be configured to monitor traffic on a single host or an entire network.

It also has a variety of add-ons and plugins that can extend its functionality.

Table of Contents

- **Key Features of Snort**
- **Basic steps to use Snort**
- **Demonstration**
- **Alert Modes in IDS/IPS**
- **Conclusion**

Key Features

- **Packet sniffing** – It can capture network packets and analyze the headers and payloads.
- **Rule-based detection** – It uses a rules file to detect malicious traffic. The rules are based on signatures of known attacks.
- **Protocol analysis** – It can analyze various protocols like TCP, UDP, IP, HTTP, FTP, SMTP, etc.
- **Preprocessors** – It has preprocessors for decoding HTTP, FTP and other protocols. Preprocessors extract useful info from the protocols before the rules engine analyzes the packets.
- **Logging and alerting** – It can log packets to the disk and generate alerts for malicious traffic.
- **Configuration** – It is highly configurable using the snort.conf file. Rules, preprocessors, outputs, etc can be configured.
- **Open source** – It is open source, free and frequently updated.

Basic Steps for Deployment

- Download and install Snort on a Linux server.
- Configure the snort.conf file as per your needs. Configure rules, preprocessors, outputs, etc.
- Start Snort in sniffing mode to analyze network traffic.
- View the alerts and logs to detect any malicious activity.
- Update the rules regularly to keep up with the latest threats.
- Fine tune the configuration to reduce false positives.
- Investigate any detected attacks and take appropriate actions

Demonstration

SNORT can be configured to run in three modes:

- Sniffer mode
- Packet Logger mode
- Network Intrusion Detection System mode

Steps of Operation (for all modes):

- To verify the installation

```
ubuntu@ip-10-10-193-28:~$ snort -V
'--> Snort! <--
Version 2.9.7.0 GRE (Build 149)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.9.1 (with TPACKET_V3)
Using PCRE version: 8.39 2016-06-14
Using ZLIB version: 1.2.11
```

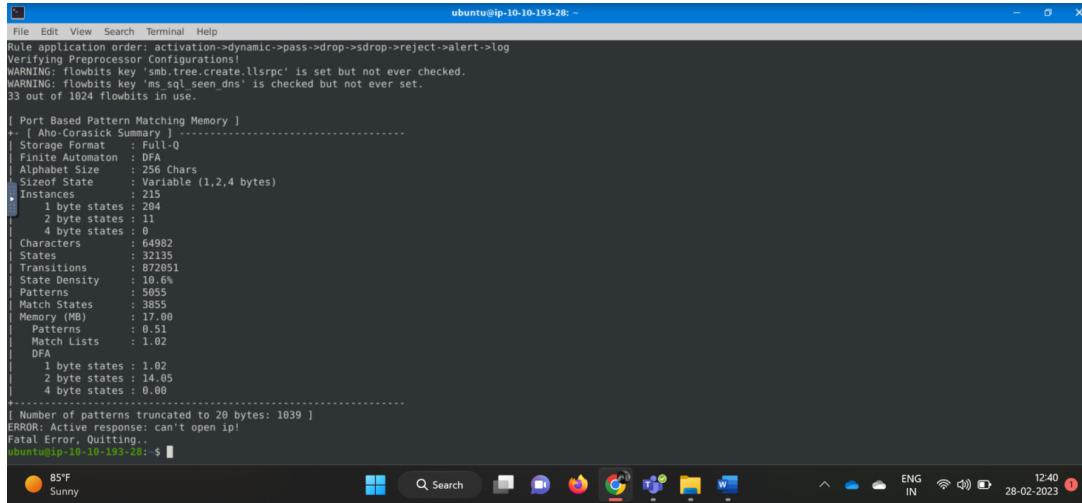
Checking the Configuration Files are Valid

In Linux configuration files of snort are stored in **/etc/snort/snort.conf**.

Snort rules are written in a language called Snort Rules Language (SRL), which is similar to a programming language.

These rules define the conditions that must be met for Snort to generate an alert. Rules can be customized to fit the specific needs of an organization or network

```
ubuntu@ip-10-10-193-28:~$ snort -c /etc/snort/snort.conf -T
Running in Test mode
==== Initializing Snort ====
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plugins!
Parsing Rules file "/etc/snort/snort.conf"
portVar 'HTTP PORTS' defined : [ 80:81 311 383 591 593 901 1220 1414 1741 1830 2381 2381 2809 3037 3128 3702 4343 4848 5250 6988 7000:7001 7144:7145 7510 7777 7779 8000 8008
+ 14 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090:9091 9443 9999 11371 34443:34444 41080 50002 55555 ]
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 : [ 80:81 110 143 311 383 591 593 901 1220 1414 1741 1830 2381 2381 2809 3037 3128 3702 4343 4848 5250 6988 7000:7001 7144:7145 7510 7777 77
79 8000 8008 8014 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090:9091 9443 9999 11371 34443:34444 41080 50002 55555 ]
portVar 'GTP PORTS' defined : [ 2123 2152 3386 ]
Detection:
  Search-Meth = AC Full O
  Split Any/Any group = enabled
  Search-Meth-Optimizations = enabled
  Maximum pattern length = 20
Tagged Packet Limit: 256
Loading dynamic engine /usr/lib/snort/dynamicengine/libsf_engine.so... done
Loading all dynamic detection libs from /usr/lib/snort/dynamicrules
Warning: No dynamic detection rules found in /usr/lib/snort/dynamicrules.
  Finished Loading all dynamic detection libs from /usr/lib/snort/dynamicrules.
Loading all dynamic preprocessor libs from /usr/lib/snort/dynamicpreprocessor/
  Loading dynamic preprocessor library /usr/lib/snort/dynamicpreprocessor//libsf_ssl_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/dynamicpreprocessor//libsf_sip_preproc.so... done
  Loading dynamic preprocessor library /usr/lib/snort/dynamicpreprocessor//libsf_imap_preproc.so... done
  
```



```
ubuntu@ip-10-10-193-20: ~
File Edit View Search Terminal Help
Rule application order: activation->dynamic->pass->drop->reject->alert->log
Verifying Preprocessor Configurations!
WARNING: flowbits key ' smb.tree.create.lllspc' is set but not ever checked.
WARNING: flowbits key 'ms_sql_seem_dns' is checked but not ever set.
33 out of 1024 flowbits in use.

[ Port Based Pattern Matching Memory ]
-- [ Aho-Corasick Summary ] -----
| Storage Format : Full-Q
| Finite Automaton : DFA
| Alphabet Size : 256 Chars
| Siz eof State Variable (1,2,4 bytes)
| Instances : 215
|   1 byte states : 204
|   2 byte states : 11
|   4 byte states : 0
| Characters : 64982
| States : 32135
| Options : 20051
| State Density : 10.6%
| Patterns : 5055
| Match States : 3855
| Memory (MB) : 17.00
| Patterns : 0.51
| Match Lists : 1.02
| DFA
|   1 byte states : 1.02
|   2 byte states : 14.85
|   4 byte states : 0.00
-----[ Number of patterns truncated to 20 bytes: 1039 ]
ERROR: Active response: can't open ip!
Fatal Error, Quitting..
ubuntu@ip-10-10-193-20: ~
```

The terminal shows the Snort configuration and memory usage statistics. It includes a warning about flowbits being set but not checked. The memory usage section details the number of states, patterns, and DFA instances.

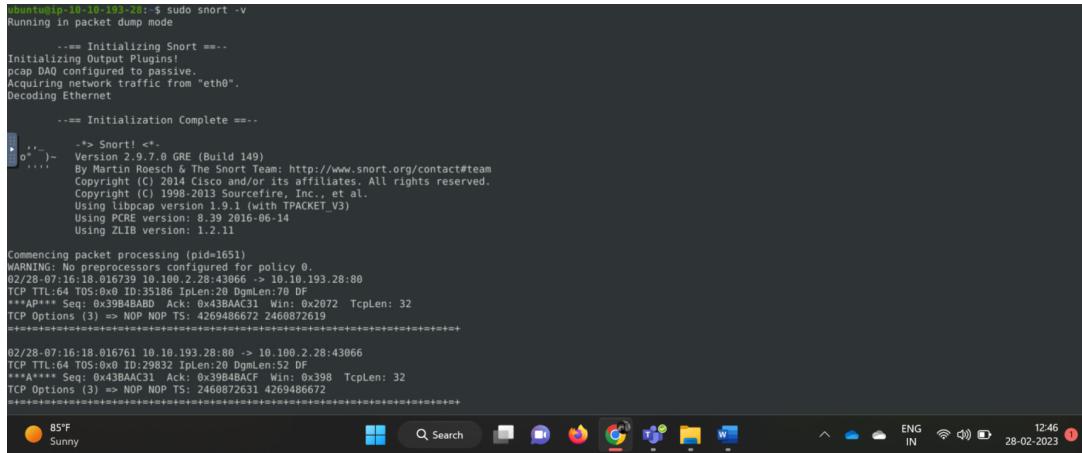
Operation Mode 1: Sniffer Mode

In Sniffer mode, it behaves like a network sniffer and captures packets passing through the network interface.

The tool displays the captured packets on the console or in a log file, allowing the user to analyze the network traffic.

This mode can be useful for network troubleshooting and monitoring, but it does not provide any intrusion detection or prevention capabilities.

sudo snort -v : Prints out the TCP/IP packets header on the screen



```
ubuntu@ip-10-10-193-20: ~$ sudo snort -v
Running in packet dump mode
    === Initializing Snort ===
Initializing Output Plugins!
scap DAO configured to passive.
Acquiring network traffic from "eth0".
Decoding Ethernet
    === Initialization Complete ===
`-'> Snort! <-
o...)- Version 2.9.7.0 GRE (Build 149)
By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.9.1 (with TPACKET_V3)
Using PCRE version: 8.39 2016-06-14
Using ZLIB version: 1.2.11

Commencing packet processing (pid=1651)
WARNING: No preprocessors configured for policy 0.
02/28/07:16:18:016739 10.100.2.28:43066 -> 10.10.193.28:80
TCP TTL:64 TOS:0x0 ID:35186 Iplen:20 DgnLen:70 DF
***AP*** Seq: 0x39B4BABD Ack: 0x43BAC31 Win: 0x2072 TcpLen: 32
TCP Options (3) => NOP NOP TS: 4269486672 2460872619
=====+
02/28/07:16:18:016761 10.10.193.28:80 -> 10.100.2.28:43066
TCP TTL:64 TOS:0x0 ID:29832 Iplen:28 DgnLen:52 DF
***A*** Seq: 0x43BAC31 Ack: 0x39B4BACF Win: 0x398 TcpLen: 32
TCP Options (3) => NOP NOP TS: 2460872631 4269486672
=====+
```

The terminal shows Snort processing network traffic in Sniffer mode. It prints out the TCP/IP packets header on the screen, including the source and destination IP addresses, port numbers, and TCP options.

```

File Edit View Search Terminal Help
ubuntu@ip-10-10-193-28: ~
IP/IPv4: 0 ( 0.000%)
IP/IPv6: 0 ( 0.000%)
IP6/IP4: 0 ( 0.000%)
IP6/IP6: 0 ( 0.000%)
GRE: 0 ( 0.000%)
GRE Eth: 0 ( 0.000%)
GRE VLAN: 0 ( 0.000%)
GRE IP4: 0 ( 0.000%)
GRE IP6: 0 ( 0.000%)
GRE PPP: 0 ( 0.000%)
GRE PPTP: 0 ( 0.000%)
GRE ARP: 0 ( 0.000%)
GRE IPX: 0 ( 0.000%)
GRE Loop: 0 ( 0.000%)
MPLS: 0 ( 0.000%)
ARP: 2 ( 0.036%)
IPX: 0 ( 0.000%)
Eth Loop: 0 ( 0.000%)
Eth Disc: 0 ( 0.000%)
IP4 Disc: 635 ( 11.571%)
IP6 Disc: 0 ( 0.000%)
TCP Disc: 0 ( 0.000%)
UDP Disc: 0 ( 0.000%)
ICMP Disc: 0 ( 0.000%)
All Discard: 635 ( 11.571%)
Bad Chk Sum: 1866 ( 33.892%)
Bad TTL: 0 ( 0.000%)
SS G 1: 0 ( 0.000%)
SS G 2: 0 ( 0.000%)
Total: 5488
=====
Snort exiting
ubuntu@ip-10-10-193-28: ~

```

85°F Sunny

sudo snort -vd : shows the TCP/IP ICMP header with application data in transmit

```

File Edit View Search Terminal Help
ubuntu@ip-10-10-193-28: ~
Running in packet dump mode

==== Initializing Snort ====
Initializing Output Plugins!
pcap DAQ configured to passive.
Acquiring network traffic from "eth0".
Decoding Ethernet

==== Initialization Complete ====

o'')~ .*> Snort! <*.
    Version 2.9.7.0 GRE (Build 149)
    By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
    Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
    Copyright (C) 1998-2013 Sourcefire, Inc., et al.
    Using libpcap version 1.9.1 (with TPACKET V3)
    Using PCRE version: 8.39 2016-06-14
    Using ZLIB version: 1.2.11

File Edit View Search Terminal Help
ubuntu@ip-10-10-193-28: ~
=====

02/28/07:17:17.219552 10.10.193.28:80 -> 10.100.2.28:43066
TCP TTL:64 TOS:0x0 ID:31021 IpLen:20 DgmLen:254 DF
***AP*** Seq: 0x43EE67C9 Ack: 0x39B5054F Win: 0x398 TcpLen: 32
TCP Options (3) => NOP NOP TS: 2460931833 4269545874
82 7E 00 C6 F8 00 00 80 00 00 01 01 00 00 ..~..... .
FF FF 00 00 01 34 00 0E 00 10 00 00 00 07 60 01 .....4.... .
1B 2E 34 36 A8 AD A7 65 6A 68 42 48 49 54 59 59 ..46...ejhbHITYY
7F 84 81 30 36 38 A0 A5 A1 5E 63 62 AD B2 AC 70 ...068...^cb...p
76 74 B7 BC B5 50 55 55 5B 60 5F 3B 41 42 57 5D vt...PUU[...;ABW]
BE C2 BB 8F 93 90 9D A2 9E 46 4C 4D D3 D7 CF \.....FLM...
8C 88 BB C0 B9 82 87 84 A7 AC A6 4E 54 54 46 .....NTTF
4B 4C 9B A0 9A 3E 42 F5 2B D8 69 DC 0C 3C 40 37 KL...>B.+i..<@7
00 19 60 DB 40 D9 1A CC 00 72 D9 D8 45 44 C1 1A ...`@...r..ED..
D8 80 F3 49 C0 62 09 18 A0 92 60 06 03 03 70 A9 ...I.b....p.
16 37 83 14 0F 98 01 06 D2 44 94 6E 74 50 02 00 .7.....D.ntP..
00 00 FF FF 00 00 00 00 00 00 00 00 00 00 FF FF FF 20 .....
F8 00 00 00 80 00 00 01 01 01 .....
```

sudo snort -X : Displays the full packet details in HEX.

Operation Mode 2: Packet Logger Mode

In Packet Logger mode, the tool logs each packet that it captures to a file for later analysis. This mode can be useful for forensic analysis or for capturing packets for offline analysis.

However, like Sniffer mode, it does not provide any intrusion detection or prevention capabilities.

Parameter “-l” – It enables the logger mode, target **log and alert** output directory. Default output folder is **/var/log/snort**. The default action is to dump as tcpdump format in **/var/log/snort**.

Starting SNORT in packet Logger Mode

```
sudo snort -dev -l .
```

//The "-l ."part of the command creates the logs in the current directory.

```
File Edit View Search Terminal Help
ubuntu@ip-10-10-193-28:~$ sudo snort -dve -l .
Running in packet logging mode

     === Initializing Snort ===
Initializing output Plugins!
Log directory set.
pcap DAO configured to passive.
Acquiring network traffic from "eth0".
Decoding Ethernet

     === Initialization Complete ===

[...]
  .-> Snort! <-
  Version 2.9.7.0 GRE (Build 149)
  By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
  Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
  Copyright (C) 1998-2013 Sourcefire, Inc., et al.
  Using libpcap version 1.9.1 (with TPACKET_V3)
  Using PCRE version 8.39 2016-06-14
  Using ZLIB version: 1.2.11

Commencing packet processing (pid=723)
(snort decoder) WARNING: IP dgm len > captured len
WARNING: No preprocessors configured for policy 0.
(snort decoder) WARNING: IP dgm len > captured len
WARNING: No preprocessors configured for policy 0.
(snort decoder) WARNING: IP dgm len > captured len
WARNING: No preprocessors configured for policy 0.
(snort decoder) WARNING: IP dgm len > captured len
WARNING: No preprocessors configured for policy 0.
```

-> Log file is created of the captured traffic.

```
ubuntu@ip-10-10-193-28: ~
File Edit View Search Terminal Help
ubuntu@ip-10-10-193-28: $ ls .
Desktop Documents Downloads Music Pictures Public Templates Videos snort.log.1677569535
ubuntu@ip-10-10-193-28: $
```

-> Next step is to read the log file generated using the command:

```
sudo snort -r <your_log_file_name>
```

// Here "-r" is Reading option to read the dumped logs in Snort.

```
ubuntu@ip-10-10-193-28: ~
File Edit View Search Terminal Help
ubuntu@ip-10-10-193-28: $ ls .
Desktop Documents Downloads Music Pictures Public Templates Videos snort.log.1677569535
ubuntu@ip-10-10-193-28: $ sudo snort -r snort.log.1677569535
Running in packet dump mode
    === Initializing Snort ===-
Initializing Output Plugins!
pcap DAO configured to read-file.
Acquiring network traffic from "snort.log.1677569535".
    === Initialization Complete ===-
o'')--> Snort!<--.
o'')--> Version 2.9.0 GRe (Build 149)
o'')--> By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
Copyright (C) 2014 Cisco and/or its affiliates. All rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using libpcap version 1.9.1 (with TPACKET_V3)
Using PCRE version 8.39 2016-06-14
Using ZLIB version 1.2.1

Commencing packet processing (pid=1820)
WARNING: No preprocessors configured for policy 0.
WARNING: No preprocessors configured for policy 0.
02/28/07:32:15.973079 IP 10.100.2.28:43584 -> 10.10.193.28:80
TCP TTL:64 TOS:0x0 IPID:56923 Iplen:20 Dgmlen:52 DF
***AP*** Seq: 0x56E147AD Ack: 0x6A9045FC Win: 0x1BB TcpLen: 32
TCP Options (3) => NOP NOP TS: 4270444631 2461830584
+=====+
WARNING: No preprocessors configured for policy 0.
02/28/07:32:15.986605 IP 10.100.2.28:43584 -> 10.10.193.28:80
TCP TTL:64 TOS:0x0 IPID:56923 Iplen:20 Dgmlen:70 DF
***AP*** Seq: 0x56E147AD Ack: 0x6A9045FC Win: 0x1BB TcpLen: 32
TCP Options (3) => NOP NOP TS: 4270444631 2461830584
+=====+
86°F Sunny 86°F Sunny 28-02-2023
```

It can read and handle the binary like output. However, if we create logs with the "**-K ASCII**" parameter, or in laymen terms, in ASCII format, Snort will not read them.

Thus to open such log files tcpdump or wireshark is needed.

Opening Log file with tcpdump

```
sudo tcpdump -r <log_file_name>
```

```
ubuntu@ip-10-10-193-28: ~
File Edit View Search Terminal Help
ubuntu@ip-10-10-193-28: $ sudo tcpdump -r snort.log.1677569535
reading from file snort.log.1677569535, link-type EN10MB (Ethernet)
07:32:15.969928 IP ip-10-10-193-28.eu-west-1.compute.internal.http > ip-10-100-2-28.eu-west-1.compute.internal.43584: Flags [P..], seq 1787828544:1787839996, ack 1457604525, win 486, options [nop,nop,T5 val 4270444646] length 11452: HTTP
07:32:15.973079 IP ip-10-100-2-28.eu-west-1.compute.internal.43584 > ip-10-10-193-28.eu-west-1.compute.internal.http: Flags [.], ack 11452, win 443, options [nop,nop,T5 val 4270444646] length 0
07:32:15.986605 IP ip-10-100-2-28.eu-west-1.compute.internal.43584 > ip-10-10-193-28.eu-west-1.compute.internal.http: Flags [P..], seq 1:19, ack 11452, win 443, options [nop,nop,T5 val 4270444646] length 18: HTTP
07:32:15.986621 IP ip-10-100-2-28.eu-west-1.compute.internal.http > ip-10-100-2-28.eu-west-1.compute.internal.43584: Flags [.], ack 19, win 486, options [nop,nop,T5 val 2461830584] length 0
07:32:16.154690 IP ip-10-100-2-28.eu-west-1.compute.internal.43584 > ip-10-10-193-28.eu-west-1.compute.internal.http: Flags [P..], seq 19:51, ack 11452, win 443, options [nop,nop,T5 val 4270444646] length 32: HTTP
07:32:16.154728 IP ip-10-100-2-28.eu-west-1.compute.internal.http > ip-10-100-2-28.eu-west-1.compute.internal.43584: Flags [.], ack 51, win 486, options [nop,nop,T5 val 2461830584] length 0
07:32:16.334108 IP ip-10-100-2-28.eu-west-1.compute.internal.43584 > ip-10-10-193-28.eu-west-1.compute.internal.http: Flags [P..], seq 51:83, ack 11452, win 443, options [nop,nop,T5 val 4270444646] length 32: HTTP
07:32:16.334129 IP ip-10-100-2-28.eu-west-1.compute.internal.http > ip-10-100-2-28.eu-west-1.compute.internal.43584: Flags [.], ack 83, win 486, options [nop,nop,T5 val 2461830584] length 0
07:32:16.336375 IP ip-10-100-2-28.eu-west-1.compute.internal.http > ip-10-100-2-28.eu-west-1.compute.internal.43584: Flags [P..], seq 11452:29350, ack 83, win 486, options [nop,nop,T5 val 4270444646] length 0
07:32:16.336404 IP ip-10-100-2-28.eu-west-1.compute.internal.http > ip-10-100-2-28.eu-west-1.compute.internal.43584: Flags [.], seq 29350:47248, ack 83, win 486, options [nop,nop,T5 val 4270444646] length 0
07:32:16.481413 IP ip-10-100-2-28.eu-west-1.compute.internal.43584 > ip-10-10-193-28.eu-west-1.compute.internal.http: Flags [P..], seq 47248:51933, ack 83, win 486, options [nop,nop,T5 val 4270444646] length 32: HTTP
07:32:16.481565 IP ip-10-100-2-28.eu-west-1.compute.internal.http > ip-10-100-2-28.eu-west-1.compute.internal.43584: Flags [.], ack 51933, win 443, options [nop,nop,T5 val 4270444646] length 0
07:32:16.481565 IP ip-10-100-2-28.eu-west-1.compute.internal.43584 > ip-10-10-193-28.eu-west-1.compute.internal.http: Flags [.], ack 83, win 486, options [nop,nop,T5 val 4270444646] length 0
07:32:16.621760 IP ip-10-100-2-28.eu-west-1.compute.internal.43584 > ip-10-10-193-28.eu-west-1.compute.internal.http: Flags [.], seq 115:177, ack 51933, win 443, options [nop,nop,T5 val 4270444646] length 0
86°F Sunny 86°F Sunny 28-02-2023
```

Operation Mode 3: IDS/IPS

In Network Intrusion Detection mode, Snort analyzes network traffic in real-time and compares it against a set of rules.

When it detects a packet that matches a rule, it generates an alert, which can be sent to a console, a log file, or an external system such as an email server or a SIEM.

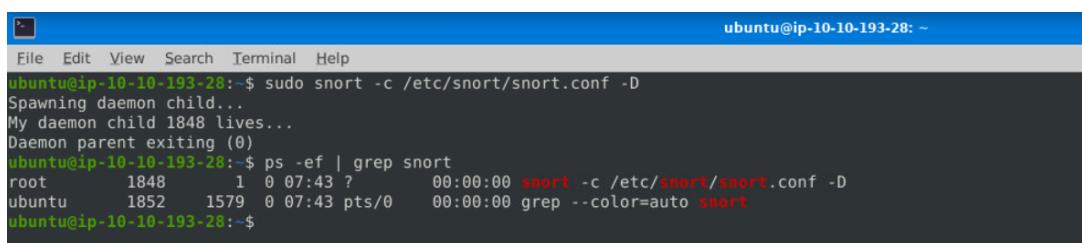
This mode provides real-time intrusion detection and prevention capabilities and is the primary mode of operation for Snort.

Parameter “-D” – This parameter is mainly used in scripts to start the Snort service in the background.

```
sudo snort -c /etc/snort/snort.conf -D

// -c : Used to define the configuration file
// -D: Background Mode.
```

Above command will start the snort instance in background and capture the traffic. Once the traffic is generated, snort will start processing the packets. Also the corresponding process can be checked with “**ps**”.



```
ubuntu@ip-10-10-193-28:~$ sudo snort -c /etc/snort/snort.conf -D
Spawning daemon child...
My daemon child 1848 lives...
Daemon parent exiting (0)
ubuntu@ip-10-10-193-28:~$ ps -ef | grep snort
root      1848  1  0 07:43 ?    00:00:00 snort -c /etc/snort/snort.conf -D
ubuntu    1852 1579  0 07:43 pts/0    00:00:00 grep --color=auto snort
ubuntu@ip-10-10-193-28:~$
```

Alert Modes in IDS/IPS

- **console** : displays alerts quickly on the console screen.
- **cmsg** : provides basic header information and payload in hexadecimal and text format.
- **full** : provides all possible information about the alert.
 - **fast** : shows only essential information such as the alert message, timestamp, source and destination IP addresses, and port numbers.
- **none** : disables alerting altogether.

To use alert modes “**-A**” parameter is used. For example to use a console alert mode below command can be used.

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

```
ubuntu@ip-10-10-193-28:~$ sudo snort -c /etc/snort/snort.conf -A console
Running in IDS mode

==== Initializing Snort ====
Initializing Output Plugins!
Initializing Preprocessors!
Initializing Plugins!
Parsing Rules file "/etc/snort/snort.conf"
PortVar 'HTTP PORTS' defined : [ 80:81 311 383 591 593 901 1220 1414 1741 1830 2381 2381 2809 3037 3128 3762 4343 4848 5250 6988 7000:7001 7144:7145 7510 7777 7779 8000 8008
8014 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090:9091 9443 9999 11371 34443:34444 41080 50002 55555 ]
PortVar 'FILE DATA PORTS' defined : [ 0:79 8165:8555 ]
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 : [ 80:81 110 143 311 383 591 593 901 1220 1414 1741 1830 2380 2381 2809 3037 3128 3762 4343 4848 5250 6988 7000:7001 7144:7145 7510 7777 7779 8000 8008
8014 8028 8080 8085 8088 8090 8118 8123 8180:8181 8243 8280 8300 8800 8888 8899 9000 9060 9080 9090:9091 9443 9999 11371 34443:34444 41080 50002 55555 ]
PortVar 'GTP PORTS' defined : [ 2123 2152 3386 ]
Detection
Search Method = AC Full 0
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 detection rules from /usr/lib/snort/dynamicroules...
Found 4151 dynamic detection rules found in directory /usr/lib/snort/dynamicroules.
Finished Loading all dynamic detection libs from /usr/lib/snort/dynamicroules.
Loading all dynamic preprocessor libs from /usr/lib/snort/dynamicpreprocessor...
Loading dynamic preprocessor library /usr/lib/snort/dynamicpreprocessor//libsf_ssl_preproc.so... done
Loading dynamic preprocessor library /usr/lib/snort/dynamicpreprocessor//libsf_sip_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_smtp_preproc.so... done
Loading dynamic preprocessor library /usr/lib/snort/dynamicpreprocessor//libsf_reputation_preproc.so... done
Location: https://www.kali.org/tutorials/network-security-hands-on/running-snort-on-kali-linux/
```

File Edit View Search Terminal Help

4151 Snort rules read

3477 detection rules

0 decoder rules

0 preprocessor rules

3477 Option Chains linked into 271 Chain Headers

0 Dynamic rules

-----[Rule Port Counts]-----

	TCP	UDP	ICMP	IP
src	151	18	0	0
dst	3306	126	0	0
any	383	48	146	22
nc	27	8	95	20
s+d	12	5	0	0

-----[detection-filter-config]-----

memory-cap : 1048576 bytes

-----[detection-filter-rules]-----

none

-----[rate-filter-config]-----

memory-cap : 1048576 bytes

-----[rate-filter-rules]-----

none

-----[event-filter-config]-----

memory-cap : 1048576 bytes

-----[event-filter-global]-----

none

-----[event-filter-local]-----

none

File Edit View Search Terminal Help

86°F Sunny

File Edit View Search Terminal Help

4151 Snort rules read

3477 detection rules

0 decoder rules

0 preprocessor rules

3477 Option Chains linked into 271 Chain Headers

0 Dynamic rules

-----[Rule Port Counts]-----

	TCP	UDP	ICMP	IP
src	151	18	0	0
dst	3306	126	0	0
any	383	48	146	22
nc	27	8	95	20
s+d	12	5	0	0

-----[detection-filter-config]-----

memory-cap : 1048576 bytes

-----[detection-filter-rules]-----

none

-----[rate-filter-config]-----

memory-cap : 1048576 bytes

-----[rate-filter-rules]-----

none

-----[event-filter-config]-----

memory-cap : 1048576 bytes

-----[event-filter-global]-----

none

-----[event-filter-local]-----

none

File Edit View Search Terminal Help

86°F Sunny

Conclusion

Snort is a powerful IDS that can help detect and prevent security threats in network traffic. It offers a range of customization options and can be used in various modes of operation.

It can detect a variety of attacks and generate alerts, logs, and block offending traffic.

Snort is a signature-based IDS that can also perform anomaly detection and has three modes of operation, sniffer, packet, and IDS/IPS.

It is free and open-source software with a large user community and is a popular choice for organizations looking to enhance their network security.

This demonstration of the instruction detection using Snort tool can be done using Windows operating system too.

Please consider **following and supporting** us to stay updated with the
latest info

Tamil S

<http://kalilinuxtutorials.com>

Tamil has a great interest in the fields of Cyber Security, OSINT, and CTF projects. Currently, he is deeply involved in researching and publishing various security tools with Kali Linux Tutorials, which is quite fascinating.

in