

SNORT

Snort is an open source IDS /IPS system founded by Martin Roesch in 1998. It is developed through the snort SourceFire company, which is the most used IDS/IPS system in the world. Cisco bought SourceFire in 2013 and integrated it with its own systems. Snort provides signature, protocol, and anomaly-based inspection to provide flexible protection against malicious attacks in general.

Snort Architecture

- packet decoder
- Detection engine and alarm
- Consists of log system

Packet Sniffer : Monitors by reading packets;

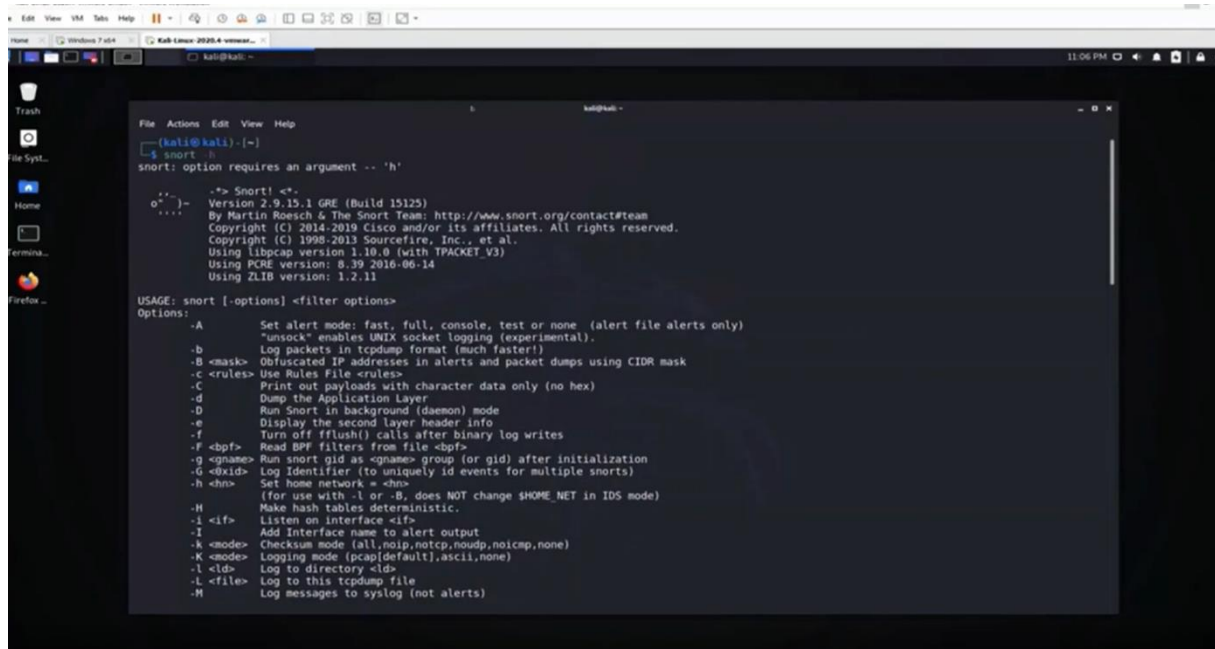
Packet Logger Mode : It saves the packets information and allows them to be written to the specified directories.

NIDS/NIPS: Provides analysis and protection on traffic according to signature database and written rule sets.

By using Snort to customize and enforce your own security rules, you can protect your environment from rapidly emerging attacks. Cisco® Talos Security Intelligence and Research Group (Talos) writes Snort rules around the clock to combat new and emerging threats. The worldwide Snort community continually provides reviews, testing, and improvements to the Snort source code. You can take advantage of the aggregated information of security teams around the world with change suggestions. Besides these benefits, snort is a very static system. In today's world, it can be difficult to keep networks dynamic and up-to-date, so its structure works by analyzing according to the rule sets in its own database.

Implementation

First, we open the help window by typing `snort -h`. In this way, we can easily see the transactions we can do.

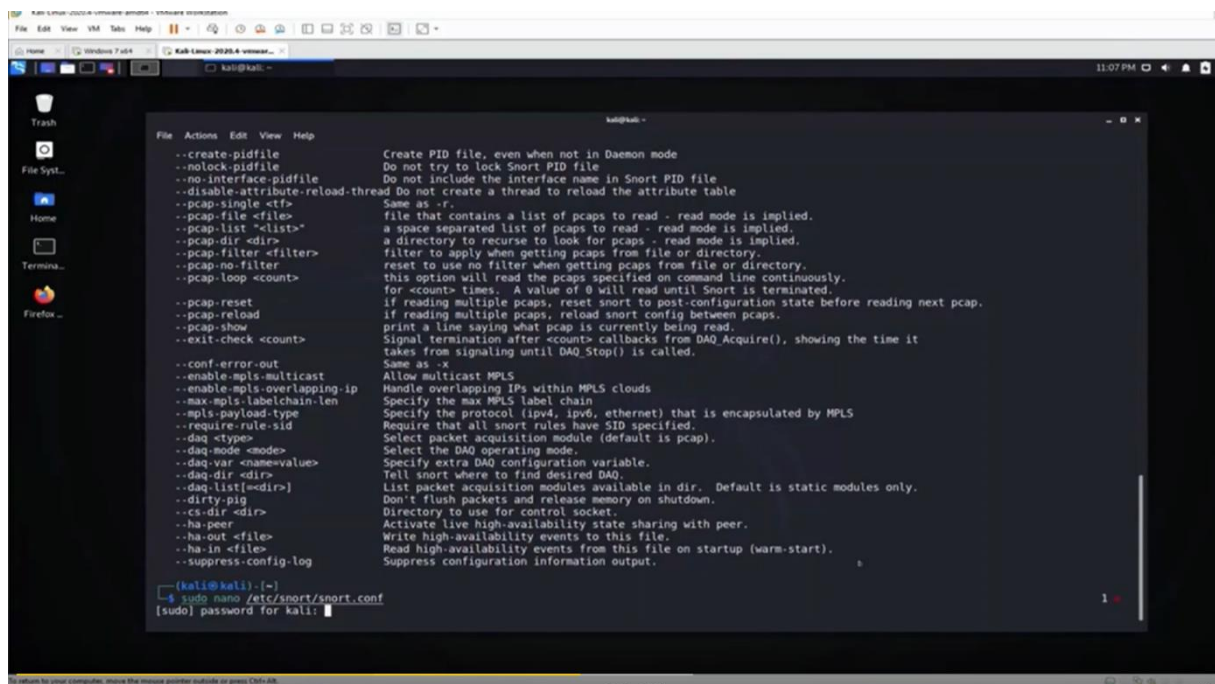


```
snort: option requires an argument -- 'h'

o*) Snort! <*)
o*) Version 2.9.15.1 GRE (Build 15125)
o*) By Martin Roesch & The Snort Team: http://www.snort.org/contact#team
o*) Copyright (C) 2014-2019 Cisco and/or its affiliates. All rights reserved.
o*) Copyright (C) 1998-2013 Sourcefire, Inc., et al.
o*) Using libpcap version 1.10.0 (with TPACKET V3)
o*) Using PCRE version: 8.39 2016-06-14
o*) Using ZLIB version: 1.2.11

USAGE: snort [-options] <filter options>
Options:
  -A      Set alert mode: fast, full, console, test or none (alert file alerts only)
          "unsock" enables UNIX socket logging (experimental).
  -b      Log packets in tcpdump format (much faster!)
  -B <mask> Obfuscated IP addresses in alerts and packet dumps using CIDR mask
  -C <rules> Use Rules File <rules>
  -C      Print out payloads with character data only (no hex)
  -d      Dump the Application Layer
  -D      Run Snort in background (daemon) mode
  -e      Display the second layer header info
  -f      Turn off fflush() calls after binary log writes
  -F <bpf> Read BPF filters from file <bpf>
  -g <gid> Run snort gid as <gid> group (or gid) after initialization
  -G <gidid> Log Identifier (to uniquely id events for multiple snorts)
  -h <hnm> Set home network = <hnm>
          (for use with -I or -B, does NOT change $HOME_NET in IDS mode)
  -H      Make hash tables deterministic.
  -i <if> Listen on interface <if>
  -I      Add Interface name to alert output
  -k <mode> Checksum mode (all,noip,notcp,noudp,noicmp,none)
  -K <mode> Logging mode (pcap[default],ascii,none)
  -L <dir> Log to directory <dir>
  -L <file> Log to this tcpdump file
  -M      Log messages to syslog (not alerts)
```

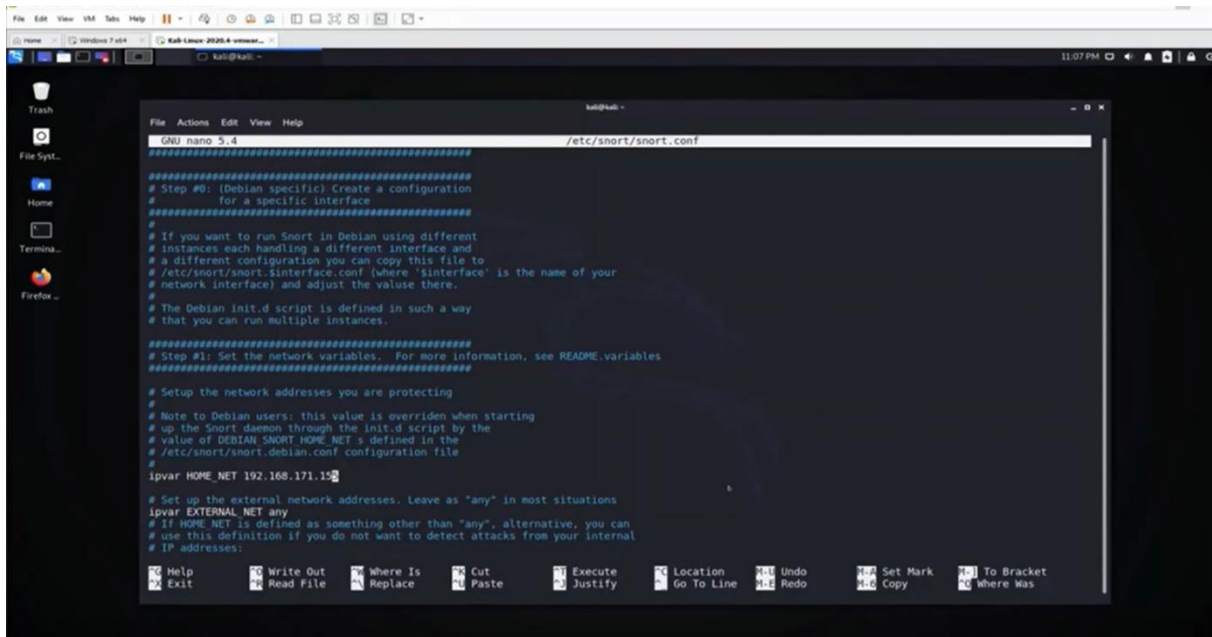
Then we call it to reach the `.Config` file and after entering the password we show us the file below



```
--create-pidfile      Create PID file, even when not in Daemon mode
--no-lock-pidfile     Do not try to lock Snort PID file
--no-interface-pidfile Do not include the interface name in Snort PID file
--disable-attribute-reload-thread Do not create a thread to reload the attribute table
--pcap-single <tf>    Same as -r.
                      file that contains a list of pcaps to read - read mode is implied.
--pcap-list <list>    a space separated list of pcaps to read - read mode is implied.
--pcap-dir <dir>      a directory to recurse to look for pcaps - read mode is implied.
--pcap-filter <filter> filter to apply when getting pcaps from file or directory.
--pcap-no-filter      reset to use no filter when getting pcaps from file or directory.
--pcap-loop <count>  this option will read the pcaps specified on command line continuously.
                      for <count> times. A value of 0 will read until Snort is terminated.
--pcap-reset          if reading multiple pcaps, reset snort to post-configuration state before reading next pcap.
--pcap-reload         print a line saying what pcap is currently being read.
--pcap-show           Signal termination after <count> callbacks from DAQ_Acquire(), showing the time it
                      takes from signaling until DAQ_Stop() is called.
--exit-check <count> Same as -x
--conf-error-out      Allow multicast MPLS
--enable-mpls-multicast Handle overlapping IPs within MPLS clouds
--enable-mpls-overlapping-ip Specify the max MPLS label chain
--max-mpls-labelchain-len Specify the protocol (ipv4, ipv6, ethernet) that is encapsulated by MPLS
--mpls-payload-type    Require that all snort rules have SID specified.
--require-rule-sid     Select packet acquisition module (default is pcap).
--daq-type             Select the DAQ operating mode.
--daq-mode <mode>      Specify extra DAQ configuration variable.
--daq-var <name=value> Tell snort where to find desired DAQ.
--daq-dir <dir>         List packet acquisition modules available in dir. Default is static modules only.
--daq-list[=dir]       Don't flush packets and release memory on shutdown.
--dirty-pig            Directory to use for control socket.
--cs-dir <dir>         Activate live high-availability state sharing with peer.
--ha-peer              Write high-availability events to this file.
--ha-out <file>        Read high-availability events from this file on startup (warm-start).
--ha-in <file>         Suppress configuration information output.
--suppress-config-log

(kali@kali)~$ sudo nano /etc/snort/snort.conf
[sudo] password for kali:
```

Next, we enter the IP of our (computer) that we want to monitor.

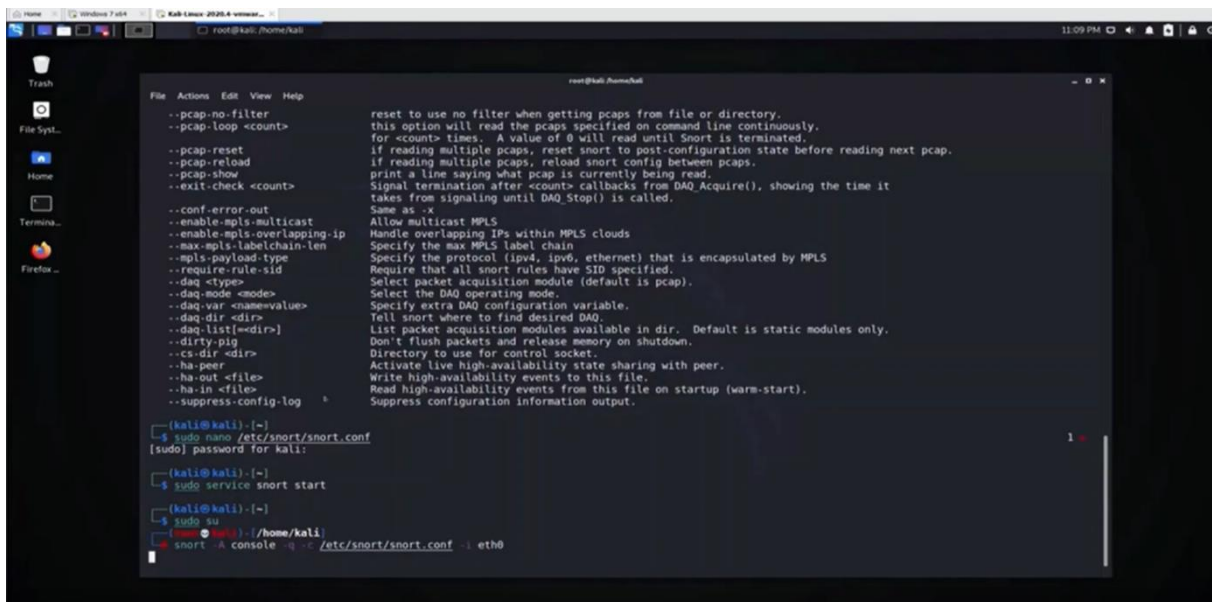


```
File Actions Edit View Help
GNU nano 5.4 /etc/snort/snort.conf

#####
# Step #0: (Debian specific) Create a configuration
# for a specific interface
#####
#
# If you want to run Snort in Debian using different
# instances each handling a different interface and
# a different configuration you can copy this file to
# /etc/snort/snort.<interface>.conf (where '<interface>' is the name of your
# network interface) and adjust the value there.
#
# The Debian init.d script is defined in such a way
# that you can run multiple instances.
#####
# Step #1: Set the network variables. For more information, see README.variables
#####
# Setup the network addresses you are protecting
#
# Note to Debian users: this value is overridden when starting
# up the Snort daemon through the init.d script by the
# value of DEBIAN_SNORT_HOME_NET defined in the
# /etc/snort/snort.debian.conf configuration file
#
ipvar HOME_NET 192.168.171.1

# Set up the external network addresses. Leave as "any" in most situations
ipvar EXTERNAL_NET any
# If HOME_NET is defined as something other than "any", alternative, you can
# use this definition if you do not want to detect attacks from your internal
# IP addresses:
#
```

We start Snort and then choose where and how we want to watch it.

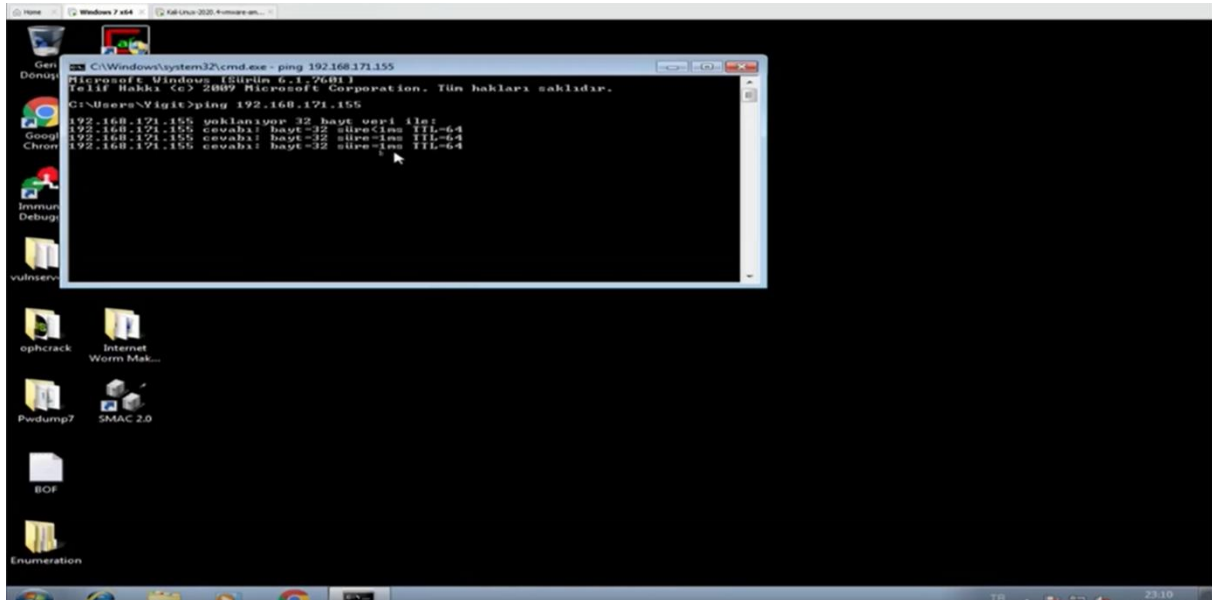


```
File Actions Edit View Help
--pcap-no-filter          reset to use no filter when getting pcaps from file or directory.
--pcap-loop <count>      this option will read the pcaps specified on command line continuously.
                          for <count> times. A value of 0 will read until Snort is terminated.
--pcap-reset              If reading multiple pcaps, reset snort to post-configuration state before reading next pcap.
--pcap-reload              If reading multiple pcaps, reload snort config between pcaps.
--pcap-show               print a line saying what pcap is currently being read.
--exit-check <count>     Signal termination after <count> callbacks from DAQ.Acquire(), showing the time it
                          takes from signaling until DAQ.Stop() is called.
                          Same as -x
--conf-error-out          Allow multicast MPLS
                          Handle overlapping IPs within MPLS clouds
--enable-mpls-multicast   Specify the max MPLS label chain
--enable-mpls-overlapping Specify the protocol (ipv4, ipv6, ethernet) that is encapsulated by MPLS
--max-mpls-labelchain-len Require that all snort rules have SID specified.
--mpls-payload-type       Select packet acquisition module (default is pcap).
--require-rule-sid        Select the DAQ operating mode.
--daq <type>              Specify extra DAQ configuration variable.
--daq-mode <mode>         Tell snort where to find desired DAQ.
--daq-var <name=value>    List packet acquisition modules available in dir. Default is static modules only.
--daq-dir <dir>           Don't flush packets and release memory on shutdown.
--daq-list[=<dir>]        Directory to use for control socket.
--dirty-pig               Activate live high-availability state sharing with peer.
--cs-dir <dir>            Write high-availability events to this file.
--ha-peer                Read high-availability events from this file on startup (warm-start).
--ha-out <file>           Suppress configuration information output.
--ha-in <file>
--suppress-config-log

(kali@kali) ~$ sudo nano /etc/snort/snort.conf
[sudo] password for kali:
(kali@kali) ~$ sudo service snort start
(kali@kali) ~$
(kali@kali) ~$ sudo su
root@kali:~/home/kali# snort -A console -q -c /etc/snort/snort.conf -i eth0
```

-A console : Show me alarm in console >> Show me Report and watch interface >> Where: eth0 , mean that its when we added IP.

We send ping to other computer's IP .



All these reports>>>

