

Autores: Carles Enric / Frederic / Diego / Toni / Julián

Actividad 1.- Snort - Suricata (IDS - ¿IPS?)

Enunciado

Un NIDS, o Sistema de Detección de Intrusiones en Red (Network Intrusion Detection System), es una tecnología destinada a supervisar y analizar el tráfico de red para detectar actividades sospechosas o anómalas que puedan indicar un intento de intrusión o ataque a un sistema informático. Estos sistemas operan al instalarse en puntos estratégicos dentro de la red donde pueden monitorear el tráfico que pasa hacia y desde todos los dispositivos en la red.

Al identificar patrones de tráfico inusuales o comportamientos maliciosos, un NIDS puede alertar a los administradores de red para tomar medidas preventivas o correctivas y así proteger la integridad de la red y sus datos

Introducción a Snort y Suricata. ¿Qué son y para qué sirven?

Snort y Suricata son sistemas de detección y prevención de intrusiones (IDS/IPS) que monitorean el tráfico de red buscando actividades maliciosas o sospechosas. Snort es uno de los más utilizados, mientras que Suricata es conocido por su capacidad de manejo de alto rendimiento y soporte multihilo.

Parte 1: Instalación y configuración en Kali Linux

Instalación de Snort y Suricata

Utiliza VirtualBox para instalar Kali Linux en una máquina virtual.

Instala Snort y Suricata siguiendo los pasos detallados en sus respectivos sitios web o guías en línea.

Parte 2: Creación de reglas básicas

- a) Regla para alerta al recibir un ping
- b) Crear una alerta si hay un ping a la 8.8.8.8
- c) Crear una alerta si se consulta Facebook
- d) Alerta por visita a un dominio por IP (lavanguardia.com)
- e) Alerta por visita al dominio lavanguardia.com
- f) Detectar peticiones HTTP GET
- g) Detectar conexiones SSH
- h) Hasta el momento estamos utilizando Snort y Suricata como IDS, intenta configurarlos como IPS

Parte 3: Práctica y evaluación

Implementa estas reglas en tu sistema IDS configurado en Kali Linux, toma capturas de pantalla de cada paso y de las alertas generadas.

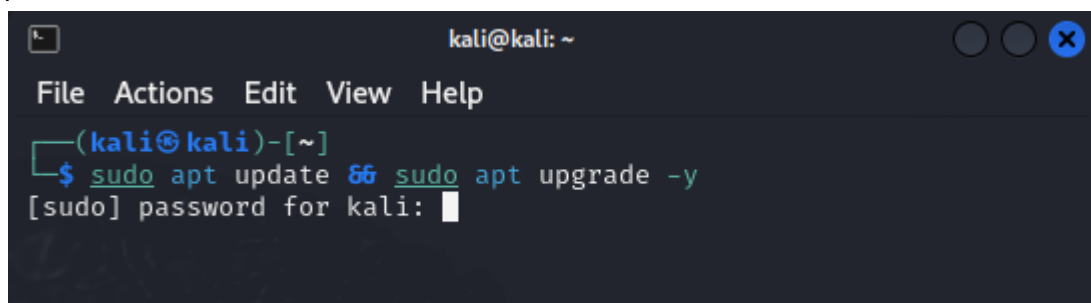
Respuesta

Para realizar esta práctica usamos como fuente el artículo de esta página web:

<https://www.zenarmor.com/docs/linux-tutorials/how-to-install-and-configure-snort-on-ubuntu-linux> aunque algunos pasos los hemos adecuado y actualizado

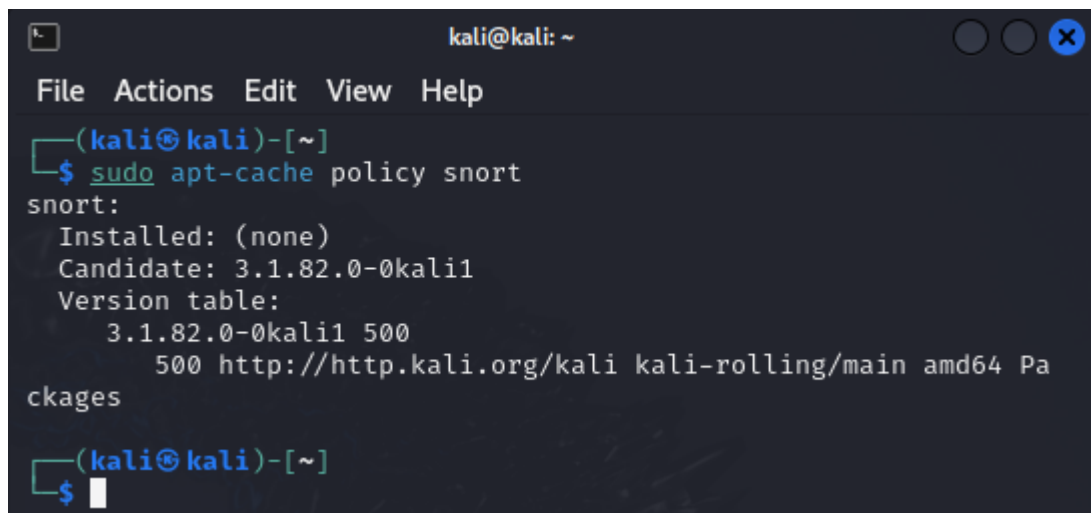
Parte 1: Instalación y configuración en Kali Linux

Usamos el comando: `sudo apt update && sudo apt upgrade -y` para actualizar la lista de paquetes disponibles e instalar las actualizaciones disponibles en el sistema de Kali Linux. Respondiendo **-y** (yes) automáticamente a cualquier pregunta que pueda surgir durante el proceso



```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ sudo apt update && sudo apt upgrade -y  
[sudo] password for kali: 
```

Comprobamos que no esté instalado snort con el comando: **sudo apt-cache policy snort**. Este comando muestra información sobre el paquete `snort` disponible en los repositorios configurados en tu sistema. Esto incluye detalles como la versión del paquete, la descripción, las dependencias, y más.



```
kali@kali: ~  
File Actions Edit View Help  
(kali@kali)-[~]  
$ sudo apt-cache policy snort  
snort:  
  Installed: (none)  
  Candidate: 3.1.82.0-0kali1  
  Version table:  
     3.1.82.0-0kali1 500  
        500 http://http.kali.org/kali kali-rolling/main amd64 Pa  
ckages  
(kali@kali)-[~]  
$ 
```

Ejecutamos el siguiente comando: **sudo apt install build-essential libpcap-dev libpcre3-dev libnet1-dev zlib1g-dev luajit hwloc libdumbnet-dev bison flex liblzma-dev openssl libssl-dev pkg-config libhwloc-dev cmake cputest libsqlite3-dev uuid-dev libcmocka-dev libnetfilter-queue-dev libmnl-dev autotools-dev liblua5.1-dev libunwind-dev libfl-dev -y** para así instalar las dependencias de paquetes necesarias para la instalación de Snort 3 que necesita para compilarse y funcionar correctamente:

build-essential: Paquetes esenciales para la compilación de software, incluyendo el compilador GCC y las herramientas de desarrollo.

libpcap-dev: Biblioteca de captura de paquetes para la monitorización de redes.

libpcre3-dev: Biblioteca de expresiones regulares compatible con Perl.

libnet1-dev: Biblioteca para la manipulación de paquetes de red.

zlib1g-dev: Biblioteca de compresión.

lua5.1-dev: Just-In-Time Compiler para el lenguaje de programación Lua.

hwloc: Biblioteca para la detección de la topología del hardware.

libdumbnet-dev: Biblioteca para el acceso de bajo nivel a la red.

bison: Generador de analizadores sintácticos.

flex: Generador de analizadores léxicos.

liblzma-dev: Biblioteca para la compresión LZMA.

openssl y libssl-dev: Bibliotecas de seguridad para la comunicación segura.

pkg-config: Herramienta para la gestión de las rutas de los archivos de configuración.

cmake: Herramienta de configuración y construcción de software.

cputest: Herramienta de prueba unitaria para C/C++.

libsqlite3-dev: Biblioteca de desarrollo para SQLite.

uuid-dev: Biblioteca de gestión de UUIDs.

libcmocka-dev: Biblioteca para la realización de pruebas unitarias en C.

libnetfilter-queue-dev: Biblioteca para la manipulación de paquetes en las colas de Netfilter.

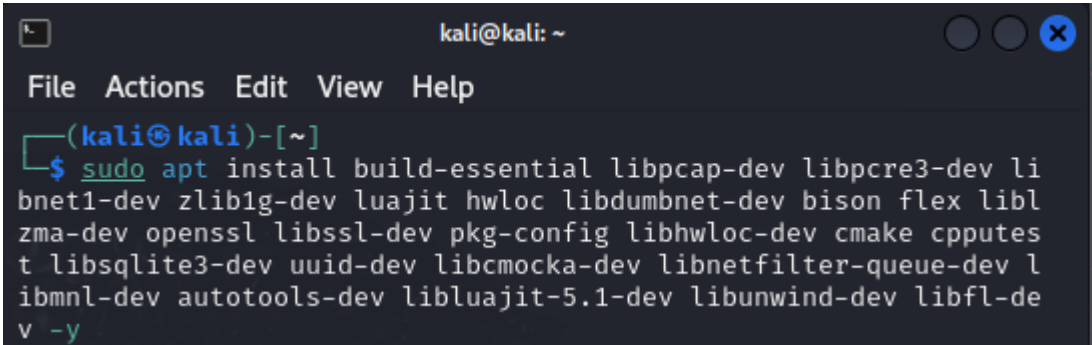
libmnl-dev: Biblioteca para la manipulación de Netlink.

autotools-dev: Herramientas para la construcción automática de software.

liblua5.1-dev: Paquete de desarrollo para LuaJIT.

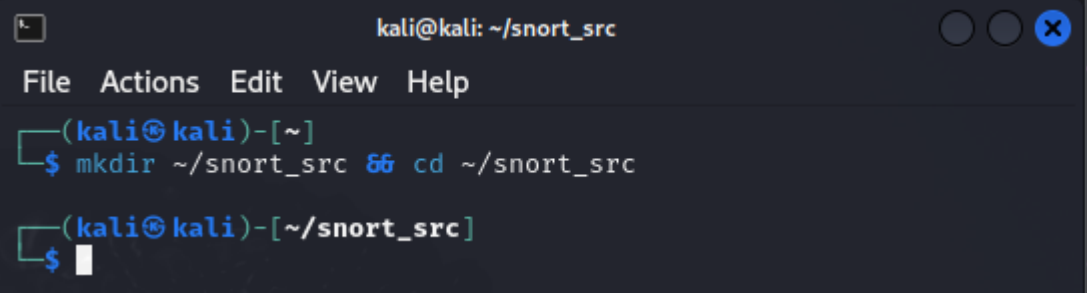
libunwind-dev: Biblioteca para el desempaquetado de la pila.

libfl-dev: Paquete de desarrollo para Flex.

A terminal window with a dark background and light text. The title bar shows 'kali@kali: ~'. The menu bar includes 'File', 'Actions', 'Edit', 'View', and 'Help'. The prompt is '(kali@kali)-[~]'. The command being entered is 'sudo apt install build-essential libpcap-dev libpcre3-dev libnet1-dev zlib1g-dev lua5.1-dev hwloc libdumbnet-dev bison flex liblzma-dev openssl libssl-dev pkg-config libhwloc-dev cmake cputest libsqlite3-dev uuid-dev libcmocka-dev libnetfilter-queue-dev libmnl-dev autotools-dev liblua5.1-dev libunwind-dev libfl-dev -y'. The command is split across multiple lines.

```
kali@kali: ~
File Actions Edit View Help
(kali@kali)-[~]
$ sudo apt install build-essential libpcap-dev libpcre3-dev li
bnet1-dev zlib1g-dev lua5.1-dev hwloc libdumbnet-dev bison flex libl
zma-dev openssl libssl-dev pkg-config libhwloc-dev cmake cputes
t libsqlite3-dev uuid-dev libcmocka-dev libnetfilter-queue-dev l
ibmnl-dev autotools-dev liblua5.1-dev libunwind-dev libfl-de
v -y
```

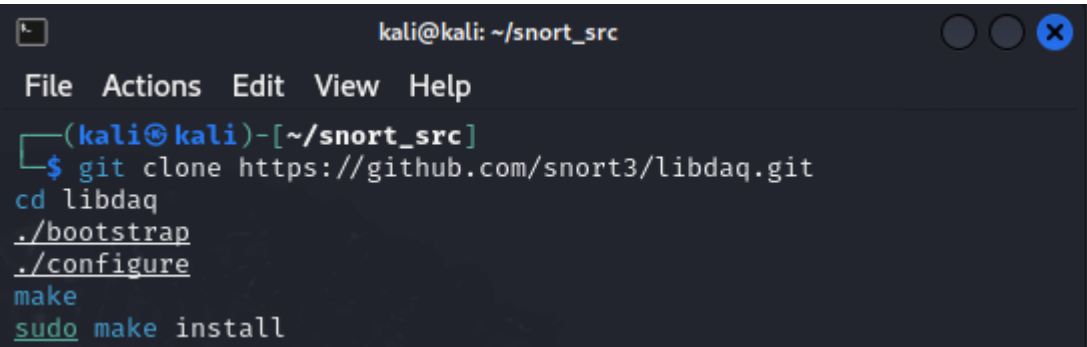
Vamos a descargar muchos archivos tarball de código fuente y otros archivos, y por eso queremos mantenerlos en una sola carpeta. Crearemos un directorio y cambiaremos su ubicación ejecutando los siguientes comandos:

A terminal window titled 'kali@kali: ~/snort_src' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(kali@kali)-[~]'. The first command is '\$ mkdir ~/snort_src && cd ~/snort_src'. The second prompt is '(kali@kali)-[~/snort_src]' with a '\$' and a cursor.

```
kali@kali: ~/snort_src
File Actions Edit View Help
(kali@kali)-[~]
$ mkdir ~/snort_src && cd ~/snort_src
(kali@kali)-[~/snort_src]
$
```

Snort DAQ (Data Acquisition library) no está incluido en los repositorios habituales. Por lo tanto, debe ser compilado e instalado desde el código fuente. Por ello descargamos e instalamos la última versión de Snort DAQ ejecutando los siguientes comandos:

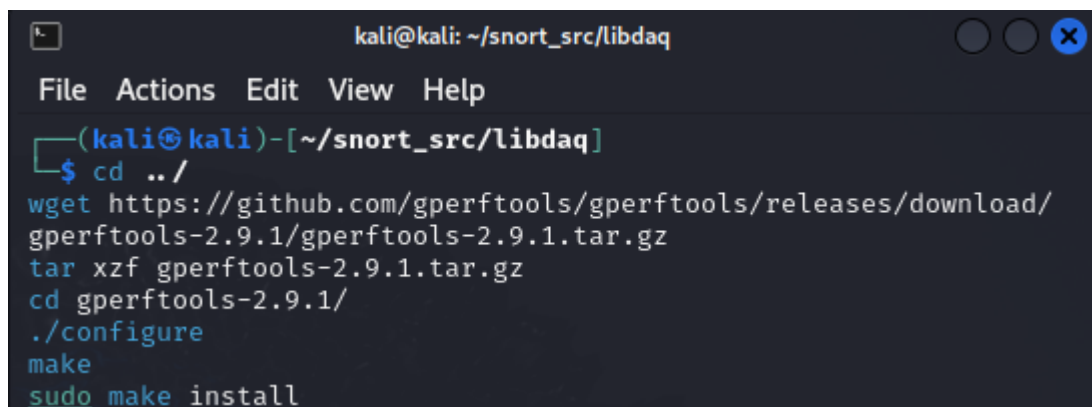
```
git clone https://github.com/snort3/libdaq.git
cd libdaq
./bootstrap
./configure
make
sudo make install
```

A terminal window titled 'kali@kali: ~/snort_src' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(kali@kali)-[~/snort_src]'. The commands are executed line by line: '\$ git clone https://github.com/snort3/libdaq.git', 'cd libdaq', './bootstrap', './configure', 'make', and 'sudo make install'.

```
kali@kali: ~/snort_src
File Actions Edit View Help
(kali@kali)-[~/snort_src]
$ git clone https://github.com/snort3/libdaq.git
cd libdaq
./bootstrap
./configure
make
sudo make install
```

Para descargar e instalar Tcmalloc, el asignador de memoria en caché por hilos de Google, un asignador de memoria optimizado para condiciones de alta concurrencia que proporcionará un rendimiento más rápido a expensas de un mayor consumo de memoria (esta es una dependencia recomendada pero opcional), ejecutamos los siguientes comandos:

```
cd ../
wget
https://github.com/gperftools/gperftools/releases/download/gperftools-2.9.1/gperftools-2.9.1.tar.gz
tar xzf gperftools-2.9.1.tar.gz
cd gperftools-2.9.1/
./configure
make
sudo make install
```

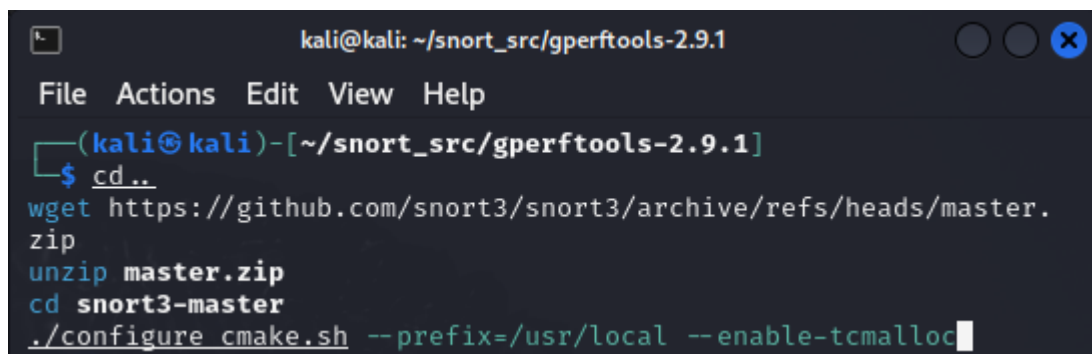
A terminal window titled 'kali@kali: ~/snort_src/libdaq' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(kali@kali)-[~/snort_src/libdaq]'. The user enters the following commands: 'cd ../', 'wget https://github.com/gperftools/gperftools/releases/download/gperftools-2.9.1/gperftools-2.9.1.tar.gz', 'tar xzf gperftools-2.9.1.tar.gz', 'cd gperftools-2.9.1/', './configure', 'make', and 'sudo make install'.

```
kali@kali: ~/snort_src/libdaq
File Actions Edit View Help
(kali@kali)-[~/snort_src/libdaq]
$ cd ../
wget https://github.com/gperftools/gperftools/releases/download/
gperftools-2.9.1/gperftools-2.9.1.tar.gz
tar xzf gperftools-2.9.1.tar.gz
cd gperftools-2.9.1/
./configure
make
sudo make install
```

Ahora que se han cumplido todos los requisitos previos, podemos descargar e instalar Snort 3 en Kali Linux.

Para descargar la versión más reciente del archivo tarball de Snort desde la página de lanzamientos, ejecutaremos los siguientes comandos:

```
cd..
wget https://github.com/snort3/snort3/archive/refs/heads/master.zip
unzip master.zip
cd snort3-master
./configure_cmake.sh --prefix=/usr/local --enable-tcmalloc
```

A terminal window titled 'kali@kali: ~/snort_src/gperftools-2.9.1' with a menu bar (File, Actions, Edit, View, Help). The prompt is '(kali@kali)-[~/snort_src/gperftools-2.9.1]'. The user enters the following commands: 'cd..', 'wget https://github.com/snort3/snort3/archive/refs/heads/master.zip', 'unzip master.zip', 'cd snort3-master', and './configure_cmake.sh --prefix=/usr/local --enable-tcmalloc'.

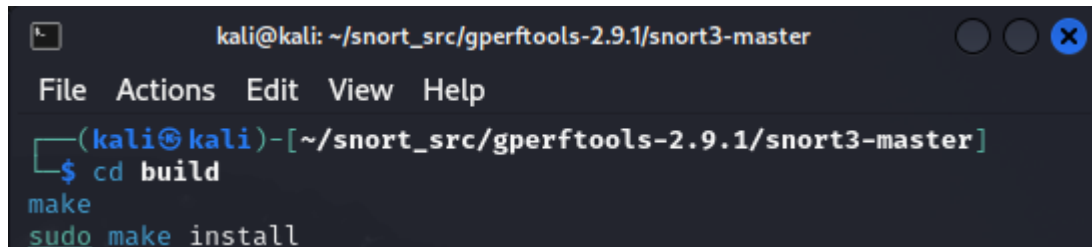
```
kali@kali: ~/snort_src/gperftools-2.9.1
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1]
$ cd..
wget https://github.com/snort3/snort3/archive/refs/heads/master.
zip
unzip master.zip
cd snort3-master
./configure_cmake.sh --prefix=/usr/local --enable-tcmalloc
```

Para navegar al directorio de compilación e instalar Snort 3 en Kali Linux antes de compilarlo, ejecutamos los siguientes comandos:

cd build

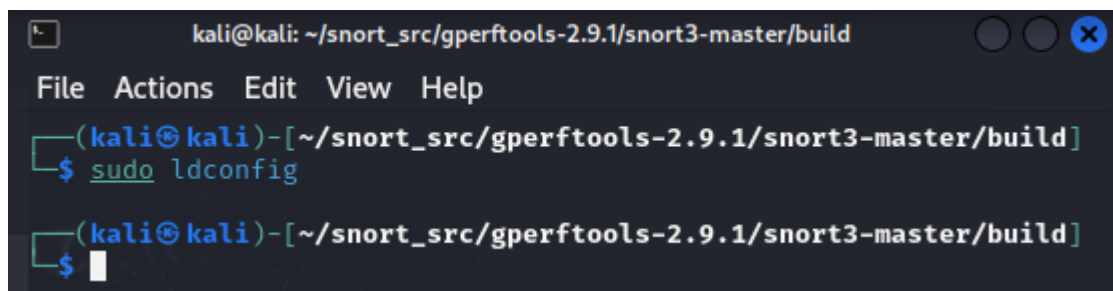
make

sudo make install



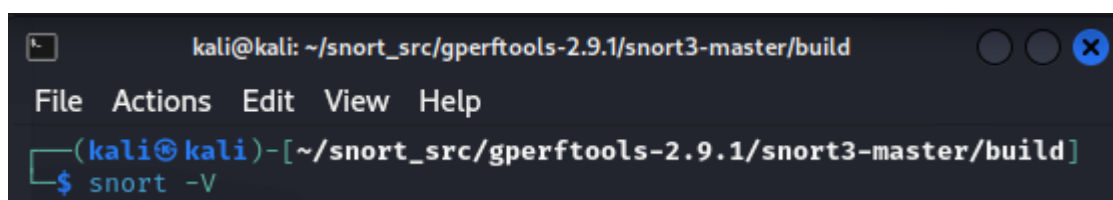
```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master]
$ cd build
make
sudo make install
```

Después de que la instalación se haya completado, actualizamos las bibliotecas compartidas ejecutando el siguiente comando: **sudo ldconfig** De lo contrario, recibiremos un error cuando intentemos ejecutar Snort



```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ sudo ldconfig
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$
```

Para verificar que Snort se ejecuta correctamente, pasamos el ejecutable de snort el indicador **-V** con el siguiente comando: **snort -V**



```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ snort -V
```

Visualizaremos en la pantalla si está correctamente instalado y en qué versión

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ snort -V

    ,,-
   o")~
    ' '

-*> Snort++ <*-
Version 3.1.84.0
By Martin Roesch & The Snort Team
http://snort.org/contact#team
Copyright (C) 2014-2024 Cisco and/or its affiliates. All
rights reserved.
Copyright (C) 1998-2013 Sourcefire, Inc., et al.
Using DAQ version 3.0.14
Using LuaJIT version 2.1.1700206165
Using OpenSSL 3.1.5 30 Jan 2024
Using libpcap version 1.10.4 (with TPACKET_V3)
Using PCRE version 8.39 2016-06-14
Using ZLIB version 1.3
Using LZMA version 5.4.5
```

Para comprobar la instalación de Snort con el archivo de configuración predeterminado, ejecutaremos el siguiente comando: **snort -c /usr/local/etc/snort/snort.lua** y nos saldrá una lista larga de datos donde nos informará que la configuración de snort está validada

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ snort -c /usr/local/etc/snort/snort.lua

o")~    Snort++ 3.1.84.0

Loading /usr/local/etc/snort/snort.lua:
Loading snort_defaults.lua:
Finished snort_defaults.lua:

pcap DAQ configured to passive.

Snort successfully validated the configuration (with 0 warnings).
o")~    Snort exiting
```

Snort tiene varias opciones para obtener más ayuda. Para ver las opciones de ayuda de Snort, podremos ejecutar el siguiente comando: **snort - -help** para averiguarlas

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ snort --help

Snort has several options to get more help:

-? list command line options (same as --help)
--help this overview of help
--help-commands [<module prefix>] output matching commands
--help-config [<module prefix>] output matching config options
--help-counts [<module prefix>] output matching peg counts
--help-limits print the int upper bounds denoted by max*
--help-module <module> output description of given module
--help-modules list all available modules with brief help
--help-modules-json dump description of all available modules in J
SON format
--help-plugins list all available plugins with brief help
--help-options [<option prefix>] output matching command line opti
ons
--help-signals dump available control signals
--list-buffers output available inspection buffers
--list-builtin [<module prefix>] output matching builtin rules
--list-gids [<module prefix>] output matching generators
--list-modules [<module type>] list all known modules
--list-plugins list all known modules
--show-plugins list module and plugin versions

--help* and --list* options preempt other processing so should be
last on the
command line since any following options are ignored. To ensure o
ptions like
--markup and --plugin-path take effect, place them ahead of the he
lp or list
options.

Options that filter output based on a matching prefix, such as --h
elp-config
won't output anything if there is no match. If no prefix is given
, everything
matches.

Report bugs to bugs@snort.org.
```


Otra posibilidad que tendremos es poder visualizar la ayuda rápida de las opciones de línea de comandos de Snort, ejecutando el siguiente comando: **snort -?**

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help

(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ snort -?
-? <option prefix> output matching command line option quick help
(same as --help-options) (optional)
-A <mode> set alert mode: none, cmg, or alert_*
-B <mask> obfuscated IP addresses in alerts and packet dumps using
  CIDR mask
-C print out payloads with character data only (no hex)
-c <conf> use this configuration
-D run Snort in background (daemon) mode
-d dump the Application Layer
-e display the second layer header info
-f turn off fflush() calls after binary log writes
-G <0xid> (same as --logid) (0:65535)
-g <gname> run snort gid as <gname> group (or gid) after initializ
ation
-H make hash tables deterministic
-h show help overview (same as --help)
-i <iface>... list of interfaces
-k <mode> checksum mode; default is all (all|noip|notcp|noudp|noic
mp|none)
-L <mode> logging mode (none, dump, pcap, or log_*)
-l <logdir> log to this directory instead of current directory
-M log messages to syslog (not alerts)
-m <umask> set the process file mode creation mask (0x000:0x1FF)
-n <count> stop after count packets (0:max53)
-O obfuscate the logged IP addresses
-Q enable inline mode operation
-q quiet mode - suppress normal logging on stdout
-R <rules> include this rules file in the default policy
-r <pcap>... (same as --pcap-list)
-s <snap> (same as --snaplen); default is 1518 (0:65535)
-T test and report on the current Snort configuration
-t <dir> chroots process to <dir> after initialization
-U use UTC for timestamps
-u <uname> run snort as <uname> or <uid> after initialization
-V (same as --version)
-v be verbose
-X dump the raw packet data starting at the link layer
-x same as --pedantic
-y include year in timestamp in the alert and log files
-z <count> maximum number of packet threads (same as --max-packet-
threads); 0 gets the number of CPU cores reported by the system; d
efault is 1 (0:max32)
--alert-before-pass evaluate alert rules before pass rules; default
is pass rules first
--bpf <filter options> are standard BPF options, as seen in TCPDum
p
--c2x output hex for given char (see also --x2c)
--create-pidfile create PID file, even when not in Daemon mode
--daq <type> select packet acquisition module (default is pcap)
--daq-batch-size <size> set the DAQ receive batch size; default is
```

Parte 2: Creación de reglas básicas

Configuración de Interfaces de Red

Las tarjetas de red modernas emplean la descarga (LRO, por ejemplo) para realizar el reensamblaje de paquetes de red en hardware, en lugar de software. En la mayoría de los casos, esto es deseable ya que reduce el estrés del sistema. Para un Sistema de Detección de Intrusiones en Red (NIDS), debemos desactivar LRO y GRO, ya que causan la truncación de paquetes más largos. Debemos establecer un servicio de systemd para modificar estos valores.

Determinaremos el nombre de la(s) interfaz(es) en la(s) que Snort escuchará ejecutando el siguiente comando: **ip address show**

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ ip address show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:29:53:62 brd ff:ff:ff:ff:ff:ff
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state down group default qlen 1000
    link/ether 08:00:27:3c:d4:39 brd ff:ff:ff:ff:ff:ff
    inet 10.0.3.15/24 brd 10.0.3.255 scope global dynamic noprefixroute
        valid_lft 69139sec preferred_lft 69139sec
    inet6 fe80::c286:e70:b429:8d2d/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

o lo determinaremos con el habitual comando: **ifconfig**. Escogeremos **eth1**

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help

(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ ifconfig
br-403c8e9bafa3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.18.0.1 netmask 255.255.0.0 broadcast 172.18.255.255
    inet6 fe80::42:e7ff:fed4:52f4 prefixlen 64 scopeid 0x20<link>
    ether 02:42:e7:da:52:f4 txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 5 bytes 550 (550.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

docker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 172.17.0.1 netmask 255.255.0.0 broadcast 172.17.255.255
    inet6 fe80::42:64ff:fea2:cecb prefixlen 64 scopeid 0x20<link>
    ether 02:42:64:a2:ce:cb txqueuelen 0 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 5 bytes 526 (526.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth0: flags=4419<UP,BROADCAST,RUNNING,PROMISC,MULTICAST> mtu 1500
    ether 08:00:27:29:53:62 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 4742 bytes 656390 (641.0 KiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

eth1: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.3.15 netmask 255.255.255.0 broadcast 10.0.3.255
    inet6 fe80::c286:e70:b429:8d2d prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:3c:d4:39 txqueuelen 1000 (Ethernet)
    RX packets 115259 bytes 122847937 (117.1 MiB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 45518 bytes 8974928 (8.5 MiB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

Estableceremos la interfaz en la que Snort está escuchando el tráfico de red en modo promiscuo para que pueda ver todo el tráfico de red transmitido a ella mediante la ejecución del siguiente comando: **sudo ip link set dev eth1 promisc on**

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help

(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ sudo ip link set dev eth1 promisc on
[sudo] password for kali:
```

Para verificar que la interfaz de red esté funcionando en modo promiscuo, ejecutaremos el siguiente comando: **ip address show eth1**

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ ip address show eth1
3: eth1: <BROADCAST,MULTICAST,PROMISC,UP,LOWER_UP> mtu 1500 qdisc fq_codel
    link/ether 08:00:27:3c:d4:39 brd ff:ff:ff:ff:ff:ff
    inet 10.0.3.15/24 brd 10.0.3.255 scope global dynamic noprefixroute
        valid_lft 68036sec preferred_lft 68036sec
    inet6 fe80::c286:e70:b429:8d2d/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

Verificamos el estado de large-receive-offload (LRO) y generic-receive-offload (GRO) para esas interfaces una vez que sepamos el nombre de la interfaz de red en la que Snort estará escuchando el tráfico mediante la ejecución del siguiente comando (reemplazando **eth1** con el nombre de nuestra interfaz) con el comando: **ethtool -k eth1 | grep receive-offload**

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ ethtool -k eth1 | grep receive-offload
generic-receive-offload: on
large-receive-offload: off [fixed]
```

Necesitamos asegurarnos de que ambos estén configurados como desactivados (o desactivados [fijo]) si vemos una salida similar a la anterior. Para desactivar la descarga de interfaz y evitar que Snort trunque paquetes grandes mayores de 1518 bytes, ejecutaremos el siguiente comando: **sudo ethtool -K eth1 gro off lro off**

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ sudo ethtool -K eth1 gro off lro off
```

Volvemos a verificar y nos cercioramos que ahora están los dos cerrados con el mismo comando anterior: **ethtool -k eth1 | grep receive-offload**

```
kali@kali: ~/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ ethtool -k eth1 | grep receive-offload
generic-receive-offload: off
large-receive-offload: off [fixed]
```


Crearemos y habilitaremos una unidad de servicio de systemd para aplicar las modificaciones de modo que sobrevivan después de un reinicio del sistema ejecutando los comandos a que escribiremos a continuación:

```
sudo su
```

```
cat > /etc/systemd/system/snort3-nic.service << 'EOL'
```

```
> [Unit]
```

```
Description=Set Snort 3 NIC in promiscuous mode and Disable GRO, LRO on boot
```

```
After=network.target
```

```
[Service]
```

```
Type=oneshot
```

```
ExecStart=/usr/sbin/ip link set dev ens18 promisc on
```

```
ExecStart=/usr/sbin/ethtool -K ens18 gro off lro off
```

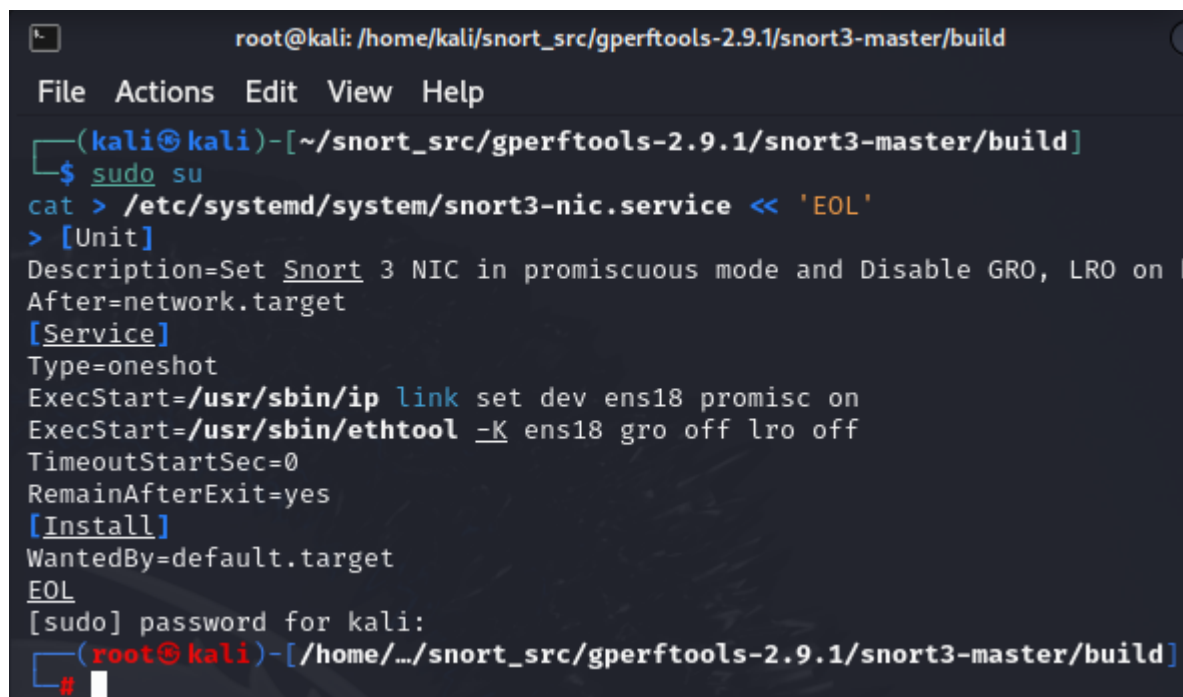
```
TimeoutStartSec=0
```

```
RemainAfterExit=yes
```

```
[Install]
```

```
WantedBy=default.target
```

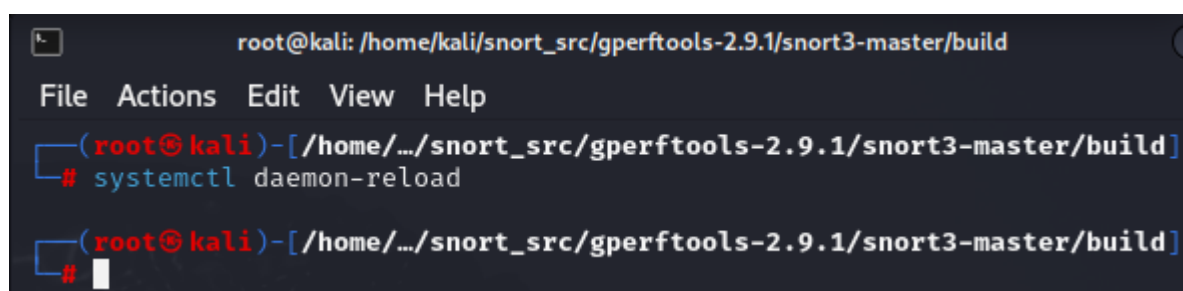
```
EOL
```

A terminal window with a dark background and light text. The title bar shows 'root@kali: /home/kali/snort_src/gperftools-2.9.1/snort3-master/build'. The terminal shows a user switching to root with 'sudo su'. Then, they use 'cat' to create a file in '/etc/systemd/system/' with the name 'snort3-nic.service'. The content of the file is entered line by line, matching the text in the previous blocks. The prompt changes from '(kali@kali)~' to '(root@kali)~'.

```
root@kali: /home/kali/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(kali@kali)~-[~/snort_src/gperftools-2.9.1/snort3-master/build]
$ sudo su
cat > /etc/systemd/system/snort3-nic.service << 'EOL'
> [Unit]
Description=Set Snort 3 NIC in promiscuous mode and Disable GRO, LRO on boot
After=network.target
[Service]
Type=oneshot
ExecStart=/usr/sbin/ip link set dev ens18 promisc on
ExecStart=/usr/sbin/ethtool -K ens18 gro off lro off
TimeoutStartSec=0
RemainAfterExit=yes
[Install]
WantedBy=default.target
EOL
[sudo] password for kali:
(root@kali)~-[~/snort_src/gperftools-2.9.1/snort3-master/build]
```

Para recargar los ajustes de configuración de systemd, ejecutaremos el siguiente comando:

systemctl daemon-reload. Este comando recargará los ajustes de configuración de systemd para que cualquier cambio realizado en las unidades de servicio, como la que acabamos de crear, se tenga en cuenta sin necesidad de reiniciar el servicio systemd

A terminal window with a dark background and light text. The title bar shows 'root@kali: /home/kali/snort_src/gperftools-2.9.1/snort3-master/build'. The terminal shows the user running 'systemctl daemon-reload'. The prompt changes from '(root@kali)~' to '(root@kali)~'.

```
root@kali: /home/kali/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(root@kali)~-[~/snort_src/gperftools-2.9.1/snort3-master/build]
# systemctl daemon-reload
(root@kali)~-[~/snort_src/gperftools-2.9.1/snort3-master/build]
#
```

Para iniciar y habilitar el servicio en el arranque, ejecutaremos el siguiente comando:
systemctl enable --now snort3-nic.service

```
root@kali: /home/kali/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(root@kali)-[/home/.../snort_src/gperftools-2.9.1/snort3-master/build]
# systemctl enable --now snort3-nic.service
```

Para ver el estado de nuestro servicio recién creado, ejecutaremos el siguiente comando:
service snort3-nic status

```
root@kali: /home/kali/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(root@kali)-[/home/.../snort_src/gperftools-2.9.1/snort3-master/build]
# service snort3-nic status
● snort3-nic.service - Set Snort 3 NIC in promiscuous mode and Disable GRO, LRO
  Loaded: loaded (/etc/systemd/system/snort3-nic.service; enabled; preset: >
  Active: active (exited) since Wed 2024-05-15 16:31:05 CEST; 8h ago
  Main PID: 727 (code=exited, status=0/SUCCESS)
  CPU: 27ms

May 15 16:31:05 kali systemd[1]: Starting snort3-nic.service - Set Snort 3 NI>
May 15 16:31:05 kali systemd[1]: Finished snort3-nic.service - Set Snort 3 NI>
```

O también podremos ejecutar el comando alternativo: **systemctl status snort3-nic**

```
root@kali: /home/kali/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
(root@kali)-[/home/.../snort_src/gperftools-2.9.1/snort3-master/build]
# systemctl status snort3-nic
● snort3-nic.service - Set Snort 3 NIC in promiscuous mode and Disable GRO, LRO o>
  Loaded: loaded (/etc/systemd/system/snort3-nic.service; enabled; preset: dis>
  Active: active (exited) since Wed 2024-05-15 16:31:05 CEST; 8h ago
  Main PID: 727 (code=exited, status=0/SUCCESS)
  CPU: 27ms

May 15 16:31:05 kali systemd[1]: Starting snort3-nic.service - Set Snort 3 NIC in>
May 15 16:31:05 kali systemd[1]: Finished snort3-nic.service - Set Snort 3 NIC in>
lines 1-8/8 (END)
```

Parte 3: Práctica y evaluación

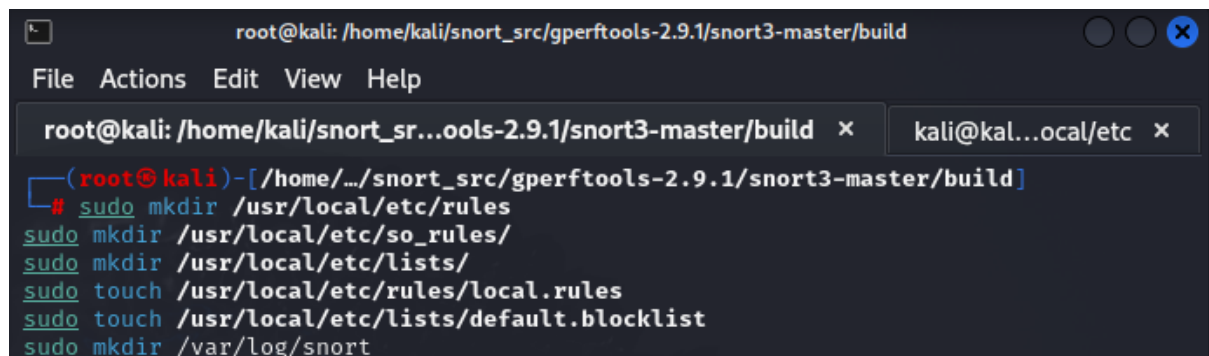
Configuración de Conjuntos de Reglas de Snort Comunitarias y Locales

Los conjuntos de reglas son el principal conducto para el motor de detección y prevención de intrusiones de Snort. Existen tres tipos de Reglas de Snort: Reglas Comunitarias, Reglas Registradas y Reglas de Suscripción.

IMPORTANTE RECORDAR: (Reiniciar snort cada vez que se introduzca nueva norma)

Crearemos algunas carpetas y archivos que Snort requiere para las reglas ejecutando los siguientes comandos:

```
sudo mkdir /usr/local/etc/rules
sudo mkdir /usr/local/etc/so_rules/
sudo mkdir /usr/local/etc/lists/
sudo touch /usr/local/etc/rules/local.rules
sudo touch /usr/local/etc/lists/default.blocklist
sudo mkdir /var/log/snort
```



```
root@kali: /home/kali/snort_src/gperftools-2.9.1/snort3-master/build
File Actions Edit View Help
root@kali: /home/kali/snort_src/gperftools-2.9.1/snort3-master/build x kali@kal...ocal/etc x
(root@kali)-[/home/.../snort_src/gperftools-2.9.1/snort3-master/build]
# sudo mkdir /usr/local/etc/rules
sudo mkdir /usr/local/etc/so_rules/
sudo mkdir /usr/local/etc/lists/
sudo touch /usr/local/etc/rules/local.rules
sudo touch /usr/local/etc/lists/default.blocklist
sudo mkdir /var/log/snort
```

Para agregar una regla que detecte el tráfico ICMP y sea excelente para validar si Snort está funcionando correctamente y produciendo alarmas. Entraremos en el archivo **local.rules** de la ruta de directorios **/usr/local/etc/rules**



```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x
(root@kali)-[/usr/local/etc/rules]
# vim local.rules
```

Pegaremos la siguiente línea en el archivo **local.rules**:

```
alert icmp any any -> any any ( msg:"ICMP Traffic Detected"; sid:10000001;
metadata:policy security-ips alert; )
```



```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x
alert icmp any any -> any any ( msg:"ICMP Traffic Detected"; sid:10000001; metadata:policy security-ips alert; )
```

Iniciaremos Snort y haremos que cargue el archivo **local.rules** (utilizando el parámetro **-R**) para asegurarnos de que estas reglas se carguen correctamente ejecutando el siguiente comando: **snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/rules/local.rules**

```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x
(root@kali)-[/usr/local/etc/rules]
# snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/rules/local.rules

o")~ Snort++ 3.1.84.0

Loading /usr/local/etc/snort/snort.lua:
Loading snort_defaults.lua:
Finished snort_defaults.lua:
```

La salida debería terminar con la siguiente línea:
Snort successfully validated the configuration (with 0 warnings).

```
pcap DAQ configured to passive.

Snort successfully validated the configuration (with 0 warnings).
o")~ Snort exiting
```

Ejecutamos Snort en modo de detección en una interfaz y registramos todas las alarmas en la consola ingresando el siguiente comando:

sudo snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/rules/local.rules -i eth1 -A alert_fast -s 65535 -k none

```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x
(root@kali)-[/usr/local/etc/rules]
# sudo snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/rules/local.rules -i eth1 -A alert_fast -s 65535 -k none

o")~ Snort++ 3.1.84.0

Loading /usr/local/etc/snort/snort.lua:
Loading snort_defaults.lua:
Finished snort_defaults.lua:
```


- c /usr/local/etc/snort/snort.lua:** El archivo de configuración snort.lua.
- R /usr/local/etc/rules/local.rules:** La ruta al archivo de reglas que contiene nuestra regla ICMP.
- i eth1:** La interfaz en la que escuchar.
- A alert_fast:** Utilizar el complemento de salida alert_fast para escribir alertas en la consola.
- s 65535:** Establecer el snaplen para que Snort no trunque ni descarte paquetes de tamaño excesivo.
- k none:** Ignorar las comprobaciones de suma de comprobación incorrectas, de lo contrario, Snort descartará paquetes con suma de comprobación incorrecta.

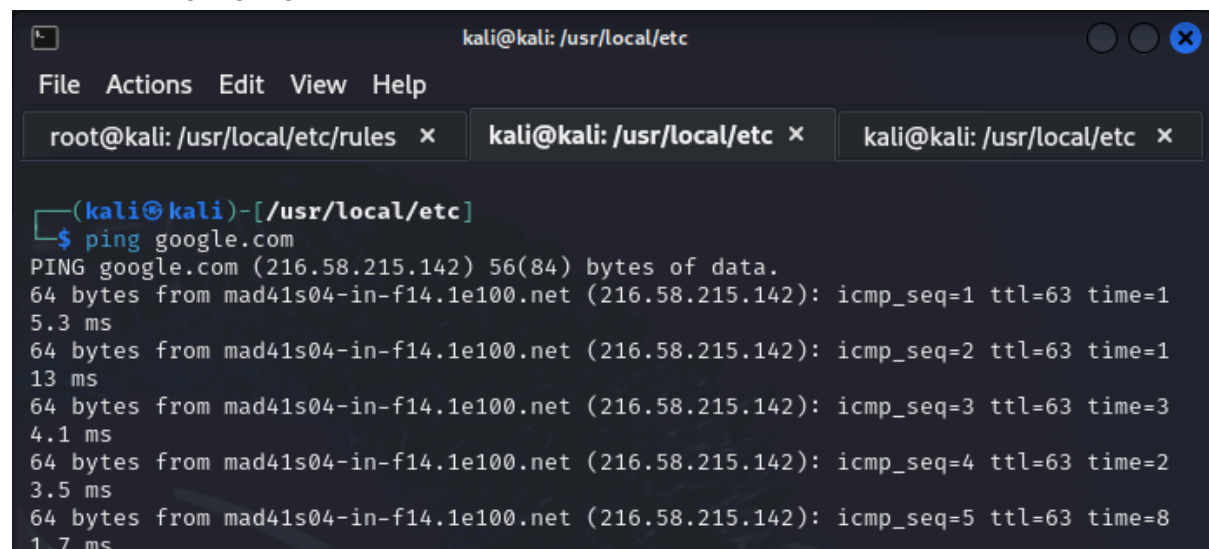
Snort cargará la configuración y luego mostrará las siguientes líneas:

```
pcap DAQ configured to passive.
Commencing packet processing
++ [0] eth1
```

Esto indica que Snort está actualmente revisando todo el tráfico en esta interfaz, eth1, con la regla cargada. Cuando el tráfico coincide con una regla, Snort generará una alerta en la consola. Ahora, desde otro ordenador, realizaremos un ping a la dirección IP de esa interfaz de Kali Linux. Deberían aparecer advertencias en la pantalla similar a las siguientes:

```
pcap DAQ configured to passive.
Commencing packet processing
++ [0] eth1
05/16-01:51:15.096631 [**] [1:10000001:0] "ICMP Traffic Detected" [**] [Priority:
0] {ICMP} 10.0.3.15 → 216.58.215.142
05/16-01:51:15.111892 [**] [1:10000001:0] "ICMP Traffic Detected" [**] [Priority:
0] {ICMP} 216.58.215.142 → 10.0.3.15
05/16-01:51:16.098271 [**] [1:10000001:0] "ICMP Traffic Detected" [**] [Priority:
0] {ICMP} 10.0.3.15 → 216.58.215.142
05/16-01:51:16.211641 [**] [1:10000001:0] "ICMP Traffic Detected" [**] [Priority:
0] {ICMP} 216.58.215.142 → 10.0.3.15
05/16-01:51:17.099616 [**] [1:10000001:0] "ICMP Traffic Detected" [**] [Priority:
0] {ICMP} 10.0.3.15 → 216.58.215.142
05/16-01:51:17.133660 [**] [1:10000001:0] "ICMP Traffic Detected" [**] [Priority:
0] {ICMP} 216.58.215.142 → 10.0.3.15
05/16-01:51:18.100239 [**] [1:10000001:0] "ICMP Traffic Detected" [**] [Priority:
0] {ICMP} 10.0.3.15 → 216.58.215.142
05/16-01:51:18.123693 [**] [1:10000001:0] "ICMP Traffic Detected" [**] [Priority:
0] {ICMP} 216.58.215.142 → 10.0.3.15
```

Haciendo ping a google.com para que fuera detectado por Snort



```
kali@kali: /usr/local/etc
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x kali@kali: /usr/local/etc x
(kali@kali)-[/usr/local/etc]
$ ping google.com
PING google.com (216.58.215.142) 56(84) bytes of data.
64 bytes from mad41s04-in-f14.1e100.net (216.58.215.142): icmp_seq=1 ttl=63 time=1
5.3 ms
64 bytes from mad41s04-in-f14.1e100.net (216.58.215.142): icmp_seq=2 ttl=63 time=1
13 ms
64 bytes from mad41s04-in-f14.1e100.net (216.58.215.142): icmp_seq=3 ttl=63 time=3
4.1 ms
64 bytes from mad41s04-in-f14.1e100.net (216.58.215.142): icmp_seq=4 ttl=63 time=2
3.5 ms
64 bytes from mad41s04-in-f14.1e100.net (216.58.215.142): icmp_seq=5 ttl=63 time=8
1.7 ms
```

Haciendo ping a nike.com para que fuera detectado por Snort

```
kali@kali: /usr/local/etc
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x kali@kali: /usr/local/etc x
$ ping nike.com
PING nike.com (18.154.22.91) 56(84) bytes of data.
64 bytes from server-18-154-22-91.mad53.r.cloudfront.net (18.154.22.91): icmp_seq=
1 ttl=63 time=14.6 ms
64 bytes from server-18-154-22-91.mad53.r.cloudfront.net (18.154.22.91): icmp_seq=
2 ttl=63 time=13.5 ms
64 bytes from server-18-154-22-91.mad53.r.cloudfront.net (18.154.22.91): icmp_seq=
3 ttl=63 time=15.3 ms
64 bytes from server-18-154-22-91.mad53.r.cloudfront.net (18.154.22.91): icmp_seq=
4 ttl=63 time=25.5 ms
64 bytes from server-18-154-22-91.mad53.r.cloudfront.net (18.154.22.91): icmp_seq=
5 ttl=63 time=14.0 ms
64 bytes from server-18-154-22-91.mad53.r.cloudfront.net (18.154.22.91): icmp_seq=
6 ttl=63 time=20.2 ms
```

Para detener Snort, presionamos Ctrl-C.

```
05/16-01:55:37.703650 [**] [1:10000001:0] "ICMP Traffic Detected" [**] [Priority:
0] {ICMP} 10.0.3.15 → 192.168.1.1
05/16-01:56:22.676731 [**] [1:10000001:0] "ICMP Traffic Detected" [**] [Priority:
0] {ICMP} 10.0.3.15 → 192.168.1.1
^C** caught int signal
= stopping
-- [0] eth1
```

```
timing
runtime: 00:06:02
seconds: 362.346798
pkts/sec: 6
o")~ Snort exiting
```

Esta es una regla útil para probar Snort, pero puede ser demasiado ruidosa para su uso en producción, así que podremos comentarla con el símbolo de almohadilla si lo deseamos. Seguidamente añadiremos otra para detectar pings específicos de 8.8.8.8 y crearemos la siguiente norma:

```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x
#alert icmp any any → any any ( msg:"ICMP Traffic Detected"; sid:10000001; metada
ta:policy security-ips alert; )
alert icmp 10.0.3.15 any → 8.8.8.8 any ( msg:"ICMP Traffic Detected IP: 8.8.8.8";
sid:10000001; metadata:policy security-ips alert; )
```

Hacemos ping a 8.8.8.8 para que sea detectado por Snort

```
kali@kali: /usr/local/etc
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x
(kali@kali)-[/usr/local/etc]
$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=63 time=17.0 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=63 time=70.9 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=63 time=23.3 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=63 time=16.4 ms
```

Volvemos a ejecutar el comando anterior y así cada vez que queramos detectar con Snort

```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x
(root@kali)-[/usr/local/etc/rules]
# sudo snort -c /usr/local/etc/snort/snort.lua -R /usr/local/etc/rules/local.rules -i eth1 -A alert_fast -s 65535 -k none

o")~ Snort++ 3.1.84.0

Loading /usr/local/etc/snort/snort.lua:
Loading snort_defaults.lua:
Finished snort_defaults.lua:
```

Detectamos 8.8.8.8 y Snort realizará el siguiente log con la norma recién creada

```
pcap DAQ configured to passive.
Commencing packet processing
++ [0] eth1
05/16-02:13:50.879116 [**] [1:10000001:0] "ICMP Traffic Detected IP: 8.8.8.8" [**]
[Priority: 0] {ICMP} 10.0.3.15 → 8.8.8.8
05/16-02:13:51.880243 [**] [1:10000001:0] "ICMP Traffic Detected IP: 8.8.8.8" [**]
[Priority: 0] {ICMP} 10.0.3.15 → 8.8.8.8
05/16-02:13:52.881928 [**] [1:10000001:0] "ICMP Traffic Detected IP: 8.8.8.8" [**]
[Priority: 0] {ICMP} 10.0.3.15 → 8.8.8.8
05/16-02:13:53.882770 [**] [1:10000001:0] "ICMP Traffic Detected IP: 8.8.8.8" [**]
[Priority: 0] {ICMP} 10.0.3.15 → 8.8.8.8
05/16-02:13:54.884274 [**] [1:10000001:0] "ICMP Traffic Detected IP: 8.8.8.8" [**]
[Priority: 0] {ICMP} 10.0.3.15 → 8.8.8.8
^C** caught int signal
= stopping
```

Ahora crearemos una alerta si se consulta Facebook en el mismo archivo **local.rules**

```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x
#alert icmp any any → any any ( msg:"ICMP Traffic Detected"; sid:10000001; metadata:policy security-ips alert; )
alert icmp 10.0.3.15 any → 8.8.8.8 any ( msg:"ICMP Traffic Detected IP: 8.8.8.8"; sid:10000001; metadata:policy security-ips alert; )
alert tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected Facebook Web"; sid:10000002; metadata:policy security-ips alert; )
~
```

Al introducir la web de Facebook.com en el navegador. Snort lo detectará y creará logs como los representados en la imagen inferior

```
pcap DAQ configured to passive.
Commencing packet processing
++ [0] eth1
05/16-02:28:09.790271 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:55714 → 34.237.73.95:443
05/16-02:28:09.790394 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:55714 → 34.237.73.95:443
05/16-02:28:13.760106 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:46110 → 34.149.100.209:443
05/16-02:28:13.777382 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:46110 → 34.149.100.209:443
05/16-02:28:13.780789 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:46110 → 34.149.100.209:443
05/16-02:28:13.800188 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:46110 → 34.149.100.209:443
05/16-02:28:13.800942 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:46110 → 34.149.100.209:443
05/16-02:28:13.800966 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:46110 → 34.149.100.209:443
```

Crearemos una alerta por visitar a un dominio por IP (lavanguardia.com) y otra alerta más por visitar al dominio lavanguardia.com

```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x kali@kali: /usr/local/etc x
#alert icmp any any → any any ( msg:"ICMP Traffic Detected"; sid:10000001; metadata:policy security-ips alert; )
alert icmp 10.0.3.15 any → 8.8.8.8 any ( msg:"ICMP Traffic Detected IP: 8.8.8.8";
sid:10000001; metadata:policy security-ips alert; )
alert tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected Facebook Web"; sid:
10000002; metadata:policy security-ips alert; )
alert icmp 10.0.3.15 any → 88.87.219.197,213.4.81.197 any ( msg:"ICMP Traffic Det
ected LaVanguardia.com"; sid:10000003; metadata:policy security-ips alert; )
alert tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected LaVanguardia.com";
sid:10000004; metadata:policy security-ips alert; )
```

Las IP's las hemos sacado de hacer ping a lavanguardia.com

```
kali@kali: /usr/local/etc
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x
(kali@kali)-[/usr/local/etc]
$ ping lavanguardia.com
PING lavanguardia.com (213.4.81.197) 56(84) bytes of data.
^C
— lavanguardia.com ping statistics —
3 packets transmitted, 0 received, 100% packet loss, time 2027ms

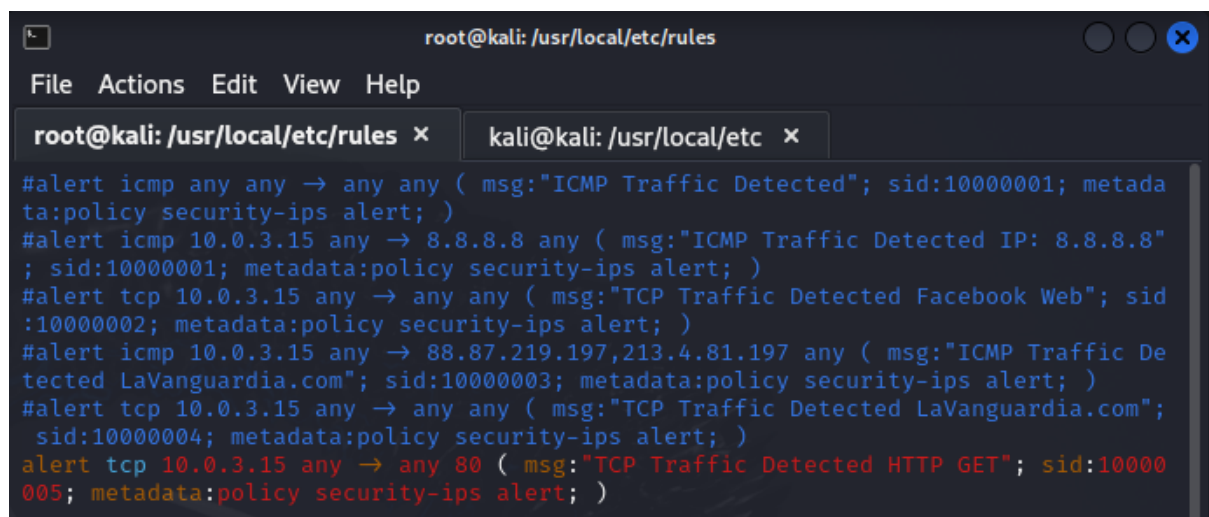
(kali@kali)-[/usr/local/etc]
$ ping lavanguardia.com
PING lavanguardia.com (88.87.219.197) 56(84) bytes of data.
^C
— lavanguardia.com ping statistics —
3 packets transmitted, 0 received, 100% packet loss, time 2031ms
```

nslookup lavanguardia.com para obtener las diferentes IPs de lavanguardia.com

Los logs de alerta que ha detectado Snort por visitar a un dominio por IP via ICMP con ping a (lavanguardia.com) y otra alerta más por visitar al dominio lavanguardia.com via TCP al navegador

```
pcap DAQ configured to passive.
Commencing packet processing
++ [0] eth1
05/16-03:16:27.384569 [**] [1:10000003:0] "ICMP Traffic Detected LaVanguardia.com"
[**] [Priority: 0] {ICMP} 10.0.3.15 → 88.87.219.197
05/16-03:16:27.389140 [**] [1:10000003:0] "ICMP Traffic Detected LaVanguardia.com"
[**] [Priority: 0] {ICMP} 10.0.3.15 → 213.4.81.197
05/16-03:16:27.503448 [**] [1:10000004:0] "TCP Traffic Detected LaVanguardia.com"
[**] [Priority: 0] {TCP} 10.0.3.15:55714 → 34.237.73.95:443
05/16-03:16:27.503448 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:55714 → 34.237.73.95:443
05/16-03:16:27.760189 [**] [1:10000004:0] "TCP Traffic Detected LaVanguardia.com"
[**] [Priority: 0] {TCP} 10.0.3.15:55714 → 34.237.73.95:443
05/16-03:16:27.760189 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:55714 → 34.237.73.95:443
05/16-03:16:28.090958 [**] [1:10000004:0] "TCP Traffic Detected LaVanguardia.com"
[**] [Priority: 0] {TCP} 10.0.3.15:46334 → 216.58.215.174:443
05/16-03:16:28.090958 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:46334 → 216.58.215.174:443
05/16-03:16:28.091048 [**] [1:10000004:0] "TCP Traffic Detected LaVanguardia.com"
[**] [Priority: 0] {TCP} 10.0.3.15:40482 → 142.250.200.110:443
05/16-03:16:28.091048 [**] [1:10000002:0] "TCP Traffic Detected Facebook Web" [**]
[Priority: 0] {TCP} 10.0.3.15:40482 → 142.250.200.110:443
```

Crearemos otra norma para detectar peticiones HTTP GET y comentaremos las otras normas por tal de poder capturar más fácilmente el log de alerta de HTTP GET

A screenshot of a terminal window showing a Snort rule configuration file. The window title is 'root@kali: /usr/local/etc/rules'. The menu bar includes 'File', 'Actions', 'Edit', 'View', and 'Help'. There are two tabs open: 'root@kali: /usr/local/etc/rules' and 'kali@kali: /usr/local/etc'. The code in the terminal shows several Snort rules for detecting ICMP and TCP traffic to specific domains and ports. The last rule is highlighted in red text: 'alert tcp 10.0.3.15 any -> any 80 (msg:"TCP Traffic Detected HTTP GET"; sid:10000005; metadata:policy security-ips alert;)'.

```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x
#alert icmp any any -> any any ( msg:"ICMP Traffic Detected"; sid:10000001; metadata:policy security-ips alert; )
#alert icmp 10.0.3.15 any -> 8.8.8.8 any ( msg:"ICMP Traffic Detected IP: 8.8.8.8"; sid:10000001; metadata:policy security-ips alert; )
#alert tcp 10.0.3.15 any -> any any ( msg:"TCP Traffic Detected Facebook Web"; sid:10000002; metadata:policy security-ips alert; )
#alert icmp 10.0.3.15 any -> 88.87.219.197,213.4.81.197 any ( msg:"ICMP Traffic Detected LaVanguardia.com"; sid:10000003; metadata:policy security-ips alert; )
#alert tcp 10.0.3.15 any -> any any ( msg:"TCP Traffic Detected LaVanguardia.com"; sid:10000004; metadata:policy security-ips alert; )
alert tcp 10.0.3.15 any -> any 80 ( msg:"TCP Traffic Detected HTTP GET"; sid:10000005; metadata:policy security-ips alert; )
```

alert tls any any -> any any (msg:"Acceso a lavanguardia.com detectado"; content:"lavanguardia.com"; tls_sni; sid:1000001; rev:1;)

Esta sería la regla adecuada para alerta sobre la visita a un dominio, sobretodo teniendo en cuenta el cifrado que hoy en día utilizan los protocolos cifrados de las consultas DNS

Para poner a prueba Snort con HTTP GET utilizaremos **curl** <https://www.amazon.com>

```
kali@kali: /usr/local/etc
File Actions Edit View Help
root@kali: /usr/local/etc/rules x kali@kali: /usr/local/etc x

(kali@kali)-[/usr/local/etc]
$ curl https://www.amazon.com
<!DOCTYPE html>
<!--[if lt IE 7]> <html lang="en-us" class="a-no-js a-lt-ie9 a-lt-ie8 a-lt-ie7"> <
![endif]-->
<!--[if IE 7]> <html lang="en-us" class="a-no-js a-lt-ie9 a-lt-ie8"> <![endif]-
->
<!--[if IE 8]> <html lang="en-us" class="a-no-js a-lt-ie9"> <![endif]-->
<!--[if gt IE 8]><!-->
<html class="a-no-js" lang="en-us"><!--<![endif]--><head>
<meta http-equiv="content-type" content="text/html; charset=UTF-8">
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1">
<title dir="ltr">Amazon.com</title>
<meta name="viewport" content="width=device-width">
<link rel="stylesheet" href="https://images-na.ssl-images-amazon.com/images/G/01/A
UIClients/AmazonUI-3c913031596ca78a3768f4e934b1cc02ce238101.secure.min._V1_.css">
<script>

if (true === true) {
    var ue_t0 = (+ new Date()),
        ue_csm = window,
        ue = { t0: ue_t0, d: function() { return (+new Date() - ue_t0); } },
        ue_furl = "fls-na.amazon.com",
        ue_mid = "ATVPDKIKX0DER",
        ue_sid = (document.cookie.match(/session-id=([0-9-]+)/) || [])[1],
        ue_sn = "opfcaptcha.amazon.com",
        ue_id = '94JVV737X2SGMKDJEMMY';
}
</script>
</head>
<body>

<!--
    To discuss automated access to Amazon data please contact api-services-sup
port@amazon.com.
    For information about migrating to our APIs refer to our Marketplace APIs
at https://developer.amazonservices.com/ref=rm_c_sv, or our Product Advertising AP
I at https://affiliate-program.amazon.com/gp/advertising/api/detail/main.html/ref=
rm_c_ac for advertising use cases.
-->

<!--
Correios.DoNotSend
-->

<div class="a-container a-padding-double-large" style="min-width:350px;padding:44p
x 0 !important">

    <div class="a-row a-spacing-double-large" style="width: 350px; margin: 0 auto">
```

Con lo que detectaremos los siguientes logs de alerta de HTTP GET realizados con curl

```
pcap DAQ configured to passive.
Commencing packet processing
++ [0] eth1
05/16-03:37:22.647940 [**] [1:10000005:0] "TCP Traffic Detected HTTP GET" [**] [Priority: 0] {TCP} 10.0.3.15:47386 → 34.107.221.82:80
05/16-03:37:22.747280 [**] [1:10000005:0] "TCP Traffic Detected HTTP GET" [**] [Priority: 0] {TCP} 10.0.3.15:47386 → 34.107.221.82:80
05/16-03:37:22.747772 [**] [1:10000005:0] "TCP Traffic Detected HTTP GET" [**] [Priority: 0] {TCP} 10.0.3.15:47386 → 34.107.221.82:80
05/16-03:37:22.747772 [**] [1:10000005:0] "TCP Traffic Detected HTTP GET" [**] [Priority: 0] {TCP} 10.0.3.15:47386 → 34.107.221.82:80
05/16-03:37:22.762890 [**] [1:10000005:0] "TCP Traffic Detected HTTP GET" [**] [Priority: 0] {TCP} 10.0.3.15:47386 → 34.107.221.82:80
```

Añadimos detecciones SSH entrantes y salientes

```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
#alert icmp any any → any any ( msg:"ICMP Traffic Detected"; sid:10000001; metadata:policy security-ips alert; )
#alert icmp 10.0.3.15 any → 8.8.8.8 any ( msg:"ICMP Traffic Detected IP: 8.8.8.8"; sid:10000001; metadata:policy security-ips alert; )
#alert tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected Facebook Web"; sid:10000002; metadata:policy security-ips alert; )
#alert icmp 10.0.3.15 any → 88.87.219.197,213.4.81.197 any ( msg:"ICMP Traffic Detected LaVanguardia.com"; sid:10000003; metadata:policy security-ips alert; )
#alert tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected LaVanguardia.com"; sid:10000004; metadata:policy security-ips alert; )
#alert tcp 10.0.3.15 any → any 80 ( msg:"TCP Traffic Detected HTTP GET"; sid:10000005; metadata:policy security-ips alert; )
alert tcp any any → any 22 ( msg:"SSH IN Traffic Detected"; sid:10000006; metadata:policy security-ips alert; )
alert tcp any 22 → any any ( msg:"SSH OUT Traffic Detected"; sid:10000006; metadata:policy security-ips alert; )
```

Probamos de conectarnos vía ssh desde el Kali Linux a un Parrot Linux (user:user / pwd:parrot / ip:192.168.10.70) para ver si detectamos la salida de ssh mediante Snort.

```
kali@kali: ~  
File Actions Edit View Help  
  
(kali㉿kali)-[~]  
$ ssh user@192.168.10.70  
user@192.168.10.70's password:  
Linux parrot 6.5.0-13parrot1-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.5.13-1  
parrot1 (2023-10-19) x86_64  
  
 _   _          _____      ___    ____\n| | | |_   _|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_\n| |_| | | | /___\\_/___\\_/___\\_/___\\_/___\\_\n| |_| | | ||_|_|_|_|_|_|_|_|_|_|_|_|_|_|_\n|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_|_  
  
The programs included with the Parrot GNU/Linux are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Parrot GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent  
permitted by applicable law.  
Last login: Sat May 18 17:31:32 2024 from 192.168.10.50  
[user@parrot]-[~]  
➔ $exit  
logout  
Connection to 192.168.10.70 closed.
```

Y detectamos los logs salientes de ssh con snort. Finalmente intentaremos configurar Snort como IPS (bloquear), ya que hasta ahora sólo lo hemos hecho como IDS (detectar/alertar)

```
pcap DAQ configured to passive.
Commencing packet processing
+- [0] eth0
05/18-19:31:25.258138 [**] [1:10000006:0] "SSH OUT Traffic Detected" [**]
  [Priority: 0] {TCP} 192.168.10.70:22 → 192.168.10.50:53848
05/18-19:31:26.061068 [**] [1:10000006:0] "SSH OUT Traffic Detected" [**]
  [Priority: 0] {TCP} 192.168.10.70:22 → 192.168.10.50:53848
05/18-19:31:26.061075 [**] [1:10000006:0] "SSH OUT Traffic Detected" [**]
  [Priority: 0] {TCP} 192.168.10.70:22 → 192.168.10.50:53848
05/18-19:31:26.061654 [**] [1:10000006:0] "SSH OUT Traffic Detected" [**]
  [Priority: 0] {TCP} 192.168.10.70:22 → 192.168.10.50:53848
05/18-19:31:26.068617 [**] [1:10000006:0] "SSH OUT Traffic Detected" [**]
  [Priority: 0] {TCP} 192.168.10.70:22 → 192.168.10.50:53848
05/18-19:31:28.826006 [**] [1:10000006:0] "SSH OUT Traffic Detected" [**]
  [Priority: 0] {TCP} 192.168.10.70:22 → 192.168.10.50:46140
05/18-19:31:28.826980 [**] [1:10000006:0] "SSH OUT Traffic Detected" [**]
  [Priority: 0] {TCP} 192.168.10.70:22 → 192.168.10.50:46140
05/18-19:31:28.850602 [**] [1:10000006:0] "SSH OUT Traffic Detected" [**]
  [Priority: 0] {TCP} 192.168.10.70:22 → 192.168.10.50:46140
05/18-19:31:29.003238 [**] [1:10000006:0] "SSH OUT Traffic Detected" [**]
  [Priority: 0] {TCP} 192.168.10.70:22 → 192.168.10.50:46140
```


Añadimos una última regla para bloquear el acceso (IPS) con el comando drop en vez de usar la opción alert que hemos usado hasta ahora

```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
#alert icmp any any → any any ( msg:"ICMP Traffic Detected"; sid:10000001
; metadata:policy security-ips alert; )
#alert icmp 10.0.3.15 any → 8.8.8.8 any ( msg:"ICMP Traffic Detected IP:
8.8.8.8"; sid:10000001; metadata:policy security-ips alert; )
#alert tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected Facebook W
eb"; sid:10000002; metadata:policy security-ips alert; )
#alert icmp 10.0.3.15 any → 88.87.219.197,213.4.81.197 any ( msg:"ICMP Tr
affic Detected LaVanguardia.com"; sid:10000003; metadata:policy security-i
ps alert; )
#alert tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected LaVanguard
ia.com"; sid:10000004; metadata:policy security-ips alert; )
#alert tcp 10.0.3.15 any → any 80 ( msg:"TCP Traffic Detected HTTP GET";
sid:10000005; metadata:policy security-ips alert; )
alert tcp any any → any 22 ( msg:"SSH IN Traffic Detected"; sid:10000006;
metadata:policy security-ips alert; )
alert tcp any 22 → any any ( msg:"SSH OUT Traffic Detected"; sid:10000006
; metadata:policy security-ips alert; )
drop tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected and Blocked"
; sid:10000007; metadata:policy security-ips alert; )
```

Usamos el mismo comando que hemos ido usando hasta ahora

```
(kali@kali)-[~]
$ sudo snort -Q -c /usr/local/etc/snort/snort.lua -i eth1 -R /usr/local
/etc/rules/local.rules -A alert_fast -s 65535 -k none

o")~ Snort++ 3.1.84.0

Loading /usr/local/etc/snort/snort.lua:
Loading snort_defaults.lua:
Finished snort_defaults.lua:
    trace
    classifications
```

Percibimos que la regla con drop ha sido detectada y ha actuado como bloqueador IPS del tráfico TCP, mediante Snort

```
pcap DAQ configured to inline.
Commencing packet processing
++ [0] eth1
05/18-22:40:35.008007 [drop] [**] [1:10000007:0] "TCP Traffic Detected an
d Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:48520 → 142.250.200.110:44
3
05/18-22:40:35.022691 [drop] [**] [1:10000007:0] "TCP Traffic Detected an
d Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:48520 → 142.250.200.110:44
3
05/18-22:40:35.027041 [drop] [**] [1:10000007:0] "TCP Traffic Detected an
d Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:48520 → 142.250.200.110:44
3
05/18-22:40:35.027291 [drop] [**] [1:10000007:0] "TCP Traffic Detected an
d Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:48520 → 142.250.200.110:44
3
05/18-22:40:35.027327 [drop] [**] [1:10000007:0] "TCP Traffic Detected an
d Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:48520 → 142.250.200.110:44
3
```

Añadimos una regla block HTTP para actuar como IPS con las páginas web HTTP

```
root@kali: /usr/local/etc/rules
File Actions Edit View Help
#alert icmp any any → any any ( msg:"ICMP Traffic Detected"; sid:10000001
; metadata:policy security-ips alert; )
#alert icmp 10.0.3.15 any → 8.8.8.8 any ( msg:"ICMP Traffic Detected IP:
8.8.8.8"; sid:10000001; metadata:policy security-ips alert; )
#alert tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected Facebook W
eb"; sid:10000002; metadata:policy security-ips alert; )
#alert icmp 10.0.3.15 any → 88.87.219.197,213.4.81.197 any ( msg:"ICMP Tr
affic Detected LaVanguardia.com"; sid:10000003; metadata:policy security-i
ps alert; )
#alert tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected LaVanguard
ia.com"; sid:10000004; metadata:policy security-ips alert; )
#alert tcp 10.0.3.15 any → any 80 ( msg:"TCP Traffic Detected HTTP GET";
sid:10000005; metadata:policy security-ips alert; )
#alert tcp any any → any 22 ( msg:"SSH IN Traffic Detected"; sid:10000006
; metadata:policy security-ips alert; )
#alert tcp any 22 → any any ( msg:"SSH OUT Traffic Detected"; sid:1000000
7; metadata:policy security-ips alert; )
#drop tcp 10.0.3.15 any → any any ( msg:"TCP Traffic Detected and Blocked
"; sid:10000008; metadata:policy security-ips alert; )
block http 10.0.3.15 any → any any ( msg:"TCP Traffic Detected and Blocke
d"; sid:10000009; metadata:policy security-ips alert; rev:1; )
```

Después de visitar un par de páginas web HTTP se generan estos logs

```
pcap DAQ configured to inline.
Commencing packet processing
++ [0] eth1
05/18-23:02:01.700217 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:43296 → 142.250.200.99:80
05/18-23:02:01.754620 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:43310 → 142.250.200.99:80
05/18-23:02:41.441523 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:36518 → 216.239.32.21:80
05/18-23:02:41.769879 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:36512 → 216.239.32.21:80
05/18-23:02:41.793532 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:36526 → 216.239.32.21:80
05/18-23:02:41.798463 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:36538 → 216.239.32.21:80
05/18-23:02:41.853865 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:50134 → 142.250.200.66:80
05/18-23:02:42.063522 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:52394 → 142.250.178.161:8
0
05/18-23:02:42.064129 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:36532 → 216.239.32.21:80
05/18-23:02:42.064224 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:36540 → 216.239.32.21:80
05/18-23:02:42.113087 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:39524 → 216.58.209.65:80
05/18-23:02:42.125059 [block] [**] [1:10000008:1] "TCP Traffic Detected a
nd Blocked" [**] [Priority: 0] {TCP} 10.0.3.15:39536 → 216.58.209.65:80
```

Se pedía una alerta al recibir un ping y alerta si se hace ping a la 8.8.8.8

En el primer caso la alerta obedecía a un ping recibido en la kali y en la segunda si cualquiera de la red interna hacía ping a la 8.8.8.8 (En algunos casos en la regla se pone IP, o se define red externa o interna en entrada y/o salida) Matices

Hay alguna anotación más por el texto

Habéis hecho un gran trabajo dado lo poco amigable que se ha puesto Kali con Snort.

Si os queréis divertir un poco menos os animo a que probéis suricata.

Yo el fichero .lua lo dejaría como configuración de origen y utilizaría un .conf para vuestras configuraciones y jugaría con ellas.

9,5/10